ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ИНСТИТУТ БИЗНЕСА»
БЕЛОРУССКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

Кафедра цифровых систем и технологий

Курсовая работа

СИСТЕМА ОРГАНИЗАЦИИ ОБУЧЕНИЯ НА ОНЛАЙН-КУРСАХ

Родионовой Алеси Олеговны
студента 4 курса группы 252
специальности «Управление
информационными ресурсами»

Научный руководитель:
Кандидат физико-математических
наук, доцент
Смалюк Антон Федорович

Минск, 2025

ГОСУДАРСТВЕННОЕ УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«ИНСТИТУТ БИЗНЕСА»
БЕЛОРУССКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА

Кафедра цифровых систем и технологий

Специальность Управление информационными ресурсами

ЗАДАНИЕ

для выполнения курсовой работы по дисциплине
«Проектирование информационных систем»

Студенту Родионова Алесе Олеговне

1. Тема проекта: Система организации обучения на онлайн-курсах.
2. Сроки сдачи проекта: 11.12.2025
3. Исходные данные к проекту

1. Назначение системы: Автоматизировать процесс управления онлайн-курсами, включая регистрацию студентов, управление курсами, загрузку учебных материалов, проверку итоговых проектов и выдачу электронных сертификатов.

2. Масштаб системы: Система будет использоваться студентами, преподавателями и администраторами образовательной платформы для организации и контроля учебного процесса.

3. Организация информационного обмена: Для хранения данных будет использована СУБД Microsoft SQL Server. Ввод исходных данных будет осуществляться с помощью различных форм и кнопок.

4. Требуемые отчетные данные и их формы: Уточнены в техническом задании.

5. Используемые технологии проектирования: Объектно-ориентированное моделирование с использованием языка UML, включающее построение диаграмм и спецификаций, описывающих концептуальную и логическую структуру проектируемой системы, а также ее статические и динамические аспекты.

4. Содержание расчетно-пояснительной записки (перечень вопросов, подлежащих разработке):

Введение.

1. Теоретические основы организации обучения на онлайн-курсах
2. Формирование требований и моделирование функциональности системы.
3. Структура системы.
4. Проектирование классов.
5. Проектирование схемы данных.
6. Разработка шаблона кода.
7. Организация тестирования и приемки информационной системы

Заключение.

Список использованных источников.

Приложения.

Задание получил	Задание выдал
Студент группы 252 УИР	Смалюк А. Ф.
Родионова А. О.	
Дата 23.09.2025	Дата 23.09.2025

ВВЕДЕНИЕ

Развитие онлайн-образования стало одним из ключевых направлений цифровой экономики. С ростом конкуренции на рынке образовательных услуг возрастает потребность в эффективных системах организации обучения, которые позволяют автоматизировать учебный процесс, обеспечивать удобный доступ к материалам и повышать вовлечённость учащихся. Для платформ онлайн-курсов особенно важно выстроить чёткую структуру взаимодействия с пользователями, поскольку качество организации обучения напрямую влияет на результаты студентов, удовлетворённость клиентов и коммерческие показатели образовательного продукта.

Современные онлайн-школы активно используют цифровые технологии для управления контентом, контроля успеваемости и коммуникации между преподавателями и учащимися. Однако при отсутствии централизованной и хорошо продуманной системы организации обучения возникают проблемы: потеря студентов на этапах прохождения курса, низкая завершённость уроков, затруднения в отслеживании прогресса, высокая нагрузка на методистов и кураторов. Ручное сопровождение, работа через разрозненные инструменты и отсутствие единого интерфейса нередко приводят к ошибкам, снижению качества сервиса и потере лояльности аудитории.

В условиях растущего рынка EdTech компании нуждаются в автоматизированных системах, которые обеспечат структурированное хранение учебных материалов, удобную навигацию по курсу, своевременную коммуникацию с пользователями, а также возможности для аналитики и контроля процесса обучения. Система организации обучения позволяет оптимизировать работу преподавателей и кураторов, уменьшить количество рутинных операций, обеспечить прозрачность учебного процесса и повысить эффективность образовательных программ.

Важным преимуществом такой системы является возможность круглосуточного доступа к контенту, адаптация под индивидуальный темп прохождения и возможность получать данные о поведении пользователей в реальном времени. Это делает процесс обучения более удобным, а работу образовательного проекта — более управляемой и масштабируемой.

Цель данной курсовой работы заключается в разработке системы организации обучения на онлайн-курсах, которая позволит структурировать учебный процесс, обеспечить удобство для пользователей и повысить эффективность работы онлайн-школы.

Объект исследования: процесс обучения на онлайн-курсах.

Предмет исследования: методы и инструменты организации и автоматизации учебного процесса в цифровой среде.

Задачи курсовой работы:

1. Закрепить теоретические знания по дисциплине «Проектирование информационных систем».
2. Проанализировать существующие подходы к организации обучения в онлайн-образовании.
3. Провести предпроектное исследование и собрать данные о требованиях к системе.
4. Разработать функциональные и нефункциональные требования к системе организации обучения.
5. Выполнить моделирование системы и разработать её концептуальную и логическую структуру.

Основным методом исследования является моделирование: построение диаграмм, описание процессов и спецификаций, характеризующих структуру и логику работы создаваемой системы.

Полученные в ходе выполнения курсовой работы знания могут быть использованы при проектировании и развитии образовательных платформ, а также при создании новых цифровых решений в сфере онлайн-обучения.

ОГЛАВЛЕНИЕ

ГЛАВА 1.
ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ОРГАНИЗАЦИИ ОБУЧЕНИЯ НА ОНЛАЙН-КУРСАХ

Разработку программного обеспечения следует начать с определения и анализа требований к будущему программному продукту и выявления бизнес-процессов, которые будут автоматизированы [1]. В случае онлайн-курсов, важно обратить внимание на процессы регистрации студентов, назначения курсов, назначения преподавателей, загрузки материалов, выполнения итоговых проектов, их оценки и выдачи сертификатов.

До внедрения информационной системы на платформе онлайн-курсов существовало несколько способов организации обучения: студенты могли проходить курсы очно, получать материалы через email или в виде файлов, а также выполнять задания вручную с последующей проверкой преподавателем. Однако эти подходы имели свои недостатки, такие как низкая автоматизация, трудности в управлении данными, отсутствие персонализированного подхода и проблемы с мониторингом успеваемости.

Внедрение автоматизированной информационной системы для организации обучения на онлайн-курсах существенно улучшит эффективность как студентов, так и преподавателей, снизив количество рутинных операций и упрощая рабочие процессы. Система позволит улучшить качество обучения, повысить вовлеченность студентов, а также ускорить процесс оценки работ и выдачи сертификатов. В конечном итоге это приведет к увеличению числа студентов, росту доходов и популяризации образовательной платформы.

Проектируемая система нацелена на автоматизацию процесса организации обучения, начиная от регистрации студента и заканчивая выдачей сертификата по завершении курса. Студент будет взаимодействовать с платформой через веб-страницу, вводя все необходимые данные, выбирая курс, выполняя задания и получая обратную связь. Преподаватели, в свою очередь, будут активно участвовать в процессе, предоставляя оценку и обратную связь по итоговым проектам студентов, а также следя за выполнением заданий.

Для более детального анализа бизнес-процесса были выделены следующие подпроцессы, которые составляют бизнес-процесс организации обучения на онлайн-курсах:

- Регистрация студента.
- Выбор курса.
- Загрузка учебных материалов.
- Назначение преподавателя.
- Выполнение итогового проекта.

- Презентация итогового проекта.
- Завершение курса и выдача сертификата.

Каждый из этих подпроцессов был представлен в диаграмме бизнес-процесса, выполненной в нотации BPMN, которая отображает последовательность действий, участников и взаимодействия с базами данных.

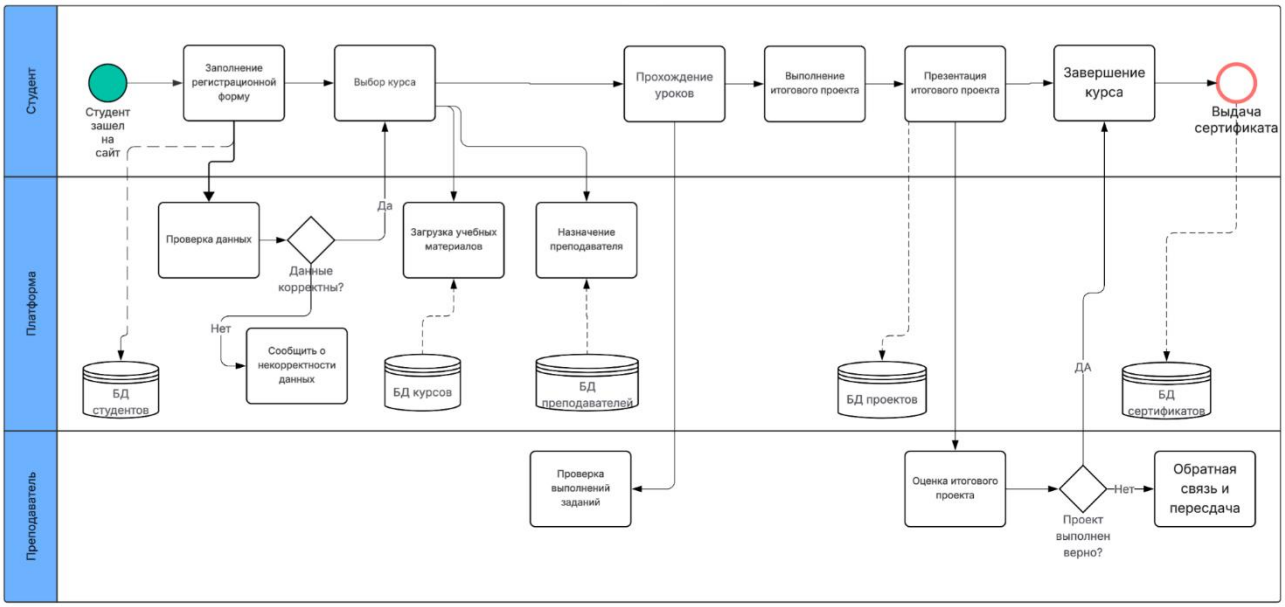

Рисунок 1 - Бизнес-процесс в нотации BPMN

Источник: собственная разработка

Описание бизнес-процесса осуществляется через последовательность шагов, которые выполняются студентом, системой и преподавателем:

1. **Пользователь заходит на сайт**: Студент открывает веб-страницу платформы онлайн-курсов.

2. **Пользователь заполняет регистрационную форму**: Студент вводит личные данные для регистрации.

3. **Система проверяет введенные данные на корректность**: После ввода данных система проверяет их на корректность. Если данные некорректны, система уведомляет пользователя и просит повторно ввести правильные данные. Если данные корректны, студент продолжает процесс.

4. **Система загружает данные в базу данных**: После успешной проверки данных система автоматически сохраняет их в базе данных (например, БД личных данных).

5. **Пользователь выбирает курс**: Студент выбирает курс, который он желает пройти.

6. **Система загружает учебные материалы**: Все материалы, необходимые для прохождения курса, загружаются и предоставляются студенту.

7. **Назначение преподавателя**: Система обращается к БД преподавателей и назначает преподавателя для выбранного курса.

8. **Студент выполняет итоговый проект**: Студент приступает к выполнению итогового проекта, который является завершающим этапом курса.

9. **Презентация итогового проекта**: Преподаватель через платформу оценивает проект студента и предоставляет обратную связь.

10. **Проверка выполнения проекта**: Если проект выполнен верно, студент переходит к завершению курса. В случае ошибок студент получает обратную связь и возможность пересдать проект.

11. **Завершение курса и получение сертификата**: После успешного выполнения проекта студент завершает курс, и система выдает сертификат.

12. **Система сохраняет информацию в базу данных**: Все данные о выполнении курса, проекте и сертификате сохраняются в соответствующие базы данных (БД проектов, БД сертификатов, БД преподавателей).

Во всех этапах бизнес-процесса активно используются базы данных, которые обеспечивают хранение и обработку информации. Студент, система и преподаватель взаимодействуют с данными, хранящимися в следующих базах данных:

- **БД личных данных**: хранит информацию о студентах (имя, email и т.д.).
- **БД курсов**: хранит информацию о курсах, их материалах и заданиях.
- **БД преподавателей**: хранит информацию о преподавателях, включая имя, специализацию и курсы, на которых они преподают.
- **БД проектов**: хранит данные об итоговых проектах студентов, их оценках и обратной связи.
- **БД сертификатов**: хранит информацию о выданных сертификатах и успешных курсах.

Все взаимодействия с данными происходят через систему, которая автоматизирует процессы загрузки, сохранения и обработки данных, что значительно упрощает работу и повышает эффективность.

Для разработки автоматизированной системы организации обучения на онлайн-курсах было сформировано техническое задание. Оно включает в себя:

- Описание функциональности системы.
- Требования к интерфейсу пользователя.
- Спецификацию баз данных.
- Сценарии взаимодействия студентов, преподавателей и администраторов.

Техническое задание содержит все необходимые параметры для разработки, тестирования и внедрения системы. Оно будет использовано для создания диаграмм вариантов использования, а также для логического проектирования системы.

Предложенная система организации обучения на онлайн-курсах позволит значительно повысить качество и доступность образовательного процесса. Она

обеспечит автоматизацию всех этапов обучения, улучшит взаимодействие студентов и преподавателей, а также упростит процесс управления курсами и оценками. В дальнейшем спроектированная модель бизнес-процесса будет использована для создания диаграмм вариантов использования и логического проектирования.

ГЛАВА 2
ФОРМИРОВАНИЕ ТРЕБОВАНИЙ И МОДЕЛИРОВАНИЕ ФУНКЦИОНАЛЬНОСТИ СИСТЕМЫ

2.1 Требования к системе

Разработка автоматизированной информационной системы организации обучения на онлайн-курсах начинается с определения требований, которым должна соответствовать создаваемая система. Все требования разделяются на **функциональные** и **нефункциональные**. Функциональные требования определяют *конкретные функции*, которые система обязана выполнять. Нефункциональные требования определяют *атрибуты качества*, связанные с производительностью, удобством использования, безопасностью и другими характеристиками программного обеспечения [2].

В рамках проектируемой информационной системы были сформированы следующие требования.

Система должна обеспечивать:

- Регистрацию новых студентов с проверкой корректности данных.
- Авторизацию и управление личным кабинетом пользователя.
- Выбор и назначение курса студентом.
- Автоматическую загрузку и предоставление учебных материалов.
- Назначение преподавателя к выбранному курсу на основе данных из БД.
- Выполнение студентом итогового проекта и загрузку его результатов в систему.
- Возможность преподавателя просматривать, оценивать и оставлять обратную связь.
- Обработку результатов проверки проекта и предоставление студенту возможности пересдачи.
- Завершение курса и автоматическую выдачу электронного сертификата.
- Ведение и обновление баз данных: студентов, курсов, преподавателей, проектов и сертификатов.
- Уведомления пользователей о важных событиях (оценка проекта, назначение преподавателя, выдача сертификата).
- Формирование отчетов по успеваемости и статусу выполнения курсов.

Диаграмма функциональных требований представлена на рисунке 2.1.

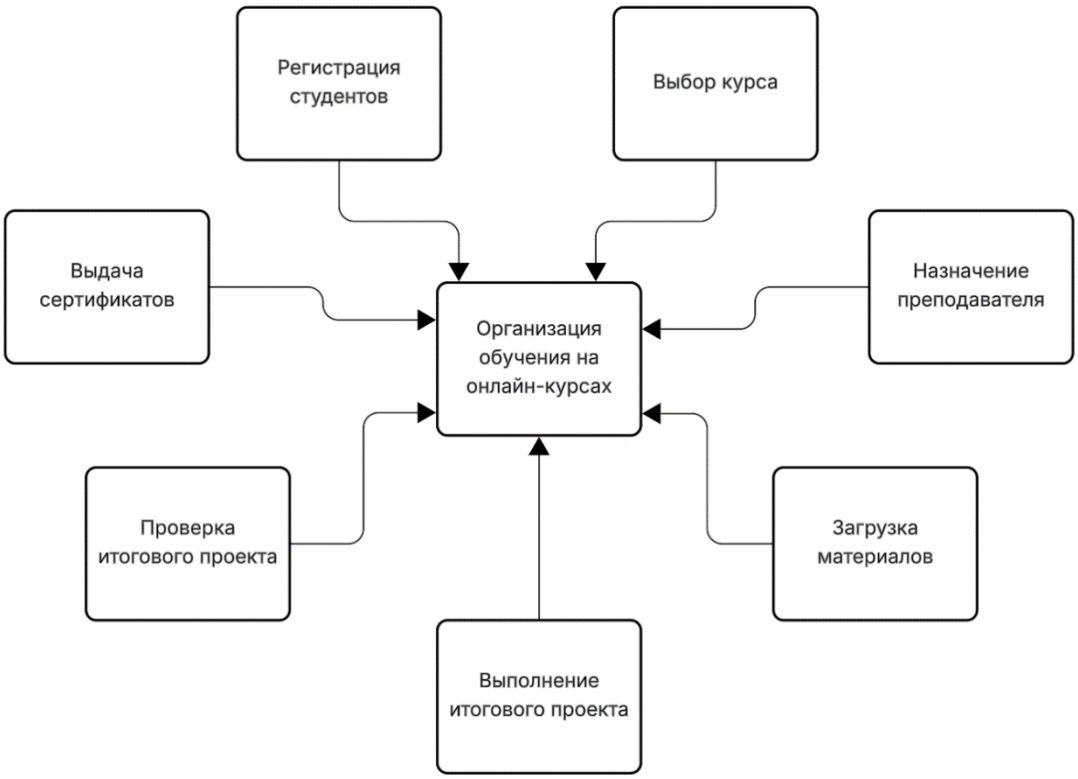

Рисунок 2.1 – Диаграмма функциональных требований

Источник: собственная разработка

К качеству работы системы предъявляются следующие требования:

● **Производительность:** система должна обеспечивать быстрое выполнение основных операций (регистрация, загрузка материалов, выдача сертификата).

● **Удобство пользования:** интерфейс должен быть интуитивно понятным для студентов и преподавателей, с минимальным количеством шагов для выполнения задач.

● **Безопасность:** защита персональных данных студентов и преподавателей, безопасное хранение файлов проектов.

● **Доступность:** интерфейс должен учитывать потребности людей со слабым зрением (контрастность, доступные размеры шрифтов).

● **Масштабируемость:** возможность увеличения числа пользователей без ухудшения производительности.

● **Многопользовательский доступ:** система должна поддерживать одновременную работу преподавателей и студентов с данными.

● **Надёжность:** система должна корректно функционировать при сбоях, сохранять данные и обеспечивать устойчивость работы.

● **Понятный логический интерфейс:** структура интерфейса должна соответствовать логике образовательного процесса.

Диаграмма нефункциональных требований представлена на рисунке 2.2.

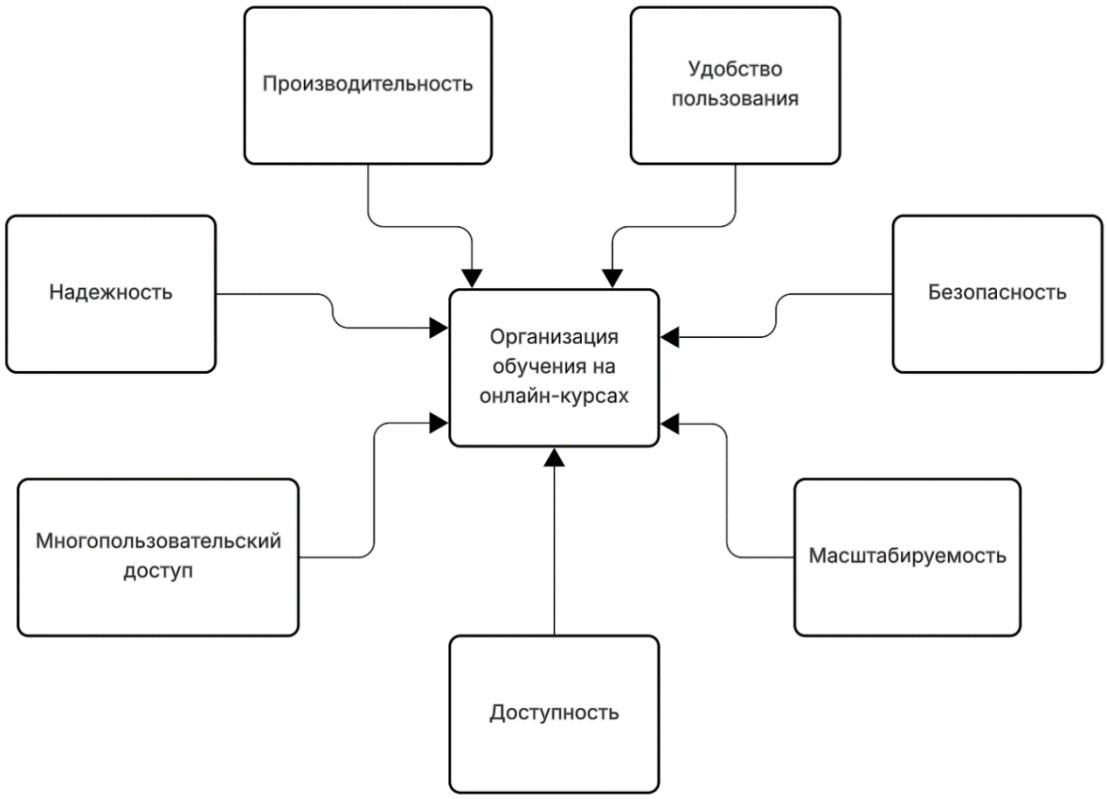

Рисунок 2.2 – Диаграмма нефункциональных требований

Источник: собственная разработка

2.2 Глоссарий системы

Необходимо было составить глоссарий системы, чтобы унифицировать терминологию, используемую при проектировании. Глоссарий разрабатываемой системы представлен в таблице 2.1.

Таблица 2.1 – Глоссарий

Термин	**Определение**
Студент	Пользователь платформы, проходящий регистрацию, выбирающий курс, выполняющий задания и итоговый проект.
Регистрационная запись	Данные, вводимые студентом при регистрации (ФИО, email, пароль и т.п.).

Продолжение таблицы 2.1

Курс	Образовательная программа, содержащая учебные материалы, задания и итоговый проект.
Учебные материалы	Электронные файлы, видеоуроки, методички и другие ресурсы, необходимые для прохождения курса.
Преподаватель	Пользователь системы, назначаемый на курс, оценивающий итоговые проекты и предоставляющий обратную связь.
Назначение преподавателя	Процесс автоматического выбора и закрепления преподавателя за студентом или курсом.
Итоговый проект	Завершающее практическое задание курса, подлежащее проверке преподавателем.
Оценка проекта	Результат проверки итогового проекта, включающий комментарии преподавателя и статус выполнения.
Статус проекта	Текущее состояние итоговой работы студента (в работе, отправлен, проверен, требуется доработка, выполнен).
Сертификат	Документ в электронном виде, выдаваемый студенту после успешного завершения курса.
База данных студентов	Хранилище информации о зарегистрированных пользователях системы.
База данных курсов	Хранилище данных о доступных курсах и их материалах.
База данных преподавателей	Информация о преподавателях, их специализациях и назначенных курсах.
База данных проектов	Хранит итоговые работы студентов, их статусы, оценки и комментарии.
База данных сертификатов	Записи о выданных студентам сертификатах.

Окончание таблицы 2.1

Уведомление	Сообщение, отправляемое студенту или преподавателю о событиях в системе (назначение преподавателя, проверка проекта, выдача сертификата).
Личный кабинет	Раздел системы, предоставляющий студентам и преподавателям доступ к персональной информации, материалам и заданиям.

Источник: собственная разработка

2.3 Диаграмма вариантов использования

Вариант использования (use case) — это последовательность действий, которые система или внешняя сущность выполняет в процессе взаимодействия с актёрами [3]. Диаграмма вариантов использования отражает множество актёров, участвующих во взаимодействии с проектируемой информационной системой онлайн-обучения. Основными элементами диаграммы являются актёр и вариант использования [4].

Варианты использования применяются для спецификации функций системы без рассмотрения её внутренней структуры. Каждый вариант использования описывает набор действий, необходимых для достижения пользователем конкретной цели. Таким образом, use case представляет собой сценарий взаимодействия пользователя с платформой онлайн-курсов.

На основе требований к системе была разработана таблица вариантов использования (таблица 2.2).

Таблица 2.2 – Описание и реализация вариантов использования информационной системы онлайн-обучения

Выход (Use Case)	**Вход (Actor)**	**Вид связи**	**Назначение**
Регистрация студента	Студент	Associate	Студент создаёт учётную запись
Вход в систему	Студент	Associate	Студент выполняет авторизацию
Выбор курса	Студент	Associate	Студент выбирает доступный курс

Продолжение таблицы 2.2

Загрузка учебных материалов	Система	Include	Система предоставляет студенту материалы курса
Назначение преподавателя	Система	Include	Система автоматически назначает преподавателя
Выполнение итогового проекта	Студент	Associate	Студент выполняет итоговое задание
Загрузка итогового проекта	Студент	Associate	Студент отправляет проект на проверку
Проверка проекта	Преподаватель	Associate	Преподаватель оценивает работу
Отправка обратной связи	Преподаватель	Include	Преподаватель оставляет комментарии и оценку
Пересдача проекта	Студент	Associate	Студент исправляет ошибки и повторно загружает работу
Завершение курса	Система	Include	Система фиксирует успешное выполнение курса
Выдача сертификата	Система	Associate	Система формирует и выдаёт сертификат
Просмотр статуса проекта	Студент	Associate	Студент отслеживает прогресс проверки
Просмотр списка студентов	Преподаватель	Associate	Преподаватель видит список студентов курса
Управление курсами	Администратор	Associate	Администратор добавляет/удаляет курсы
Управление преподавателями	Администратор	Associate	Добавление/редактирование данных преподавателей

Окончание таблицы 2.2

Управление материалами	Администратор	Associate	Добавление, удаление и обновление материалов курса

Источник: собственная разработка

Согласно таблице, для разрабатываемой системы были выделены следующие действующие актеры:

Студент — получает доступ к обучению, сдаёт задания и получает результаты.

Варианты использования:

- Регистрация
- Вход в систему
- Выбор курса
- Просмотр материалов
- Загрузка итогового проекта
- Просмотр статуса проверки
- Пересдача проекта
- Получение сертификата

Преподаватель — отвечает за проверку итоговых проектов и взаимодействие со студентами.

Варианты использования:

- Вход в систему
- Просмотр списка студентов
- Проверка итогового проекта
- Отправка обратной связи

Администратор — управляет структурой и содержанием обучающей платформы.

Варианты использования:

- Вход в систему
- Управление курсами
- Управление учебными материалами
- Управление преподавателями

Система (автоматизированные процессы) — выполняет действия, не требующие участия пользователя.

Исполняемые функции:

- Загрузка учебных материалов
- Назначение преподавателя
- Завершение курса

- Выдача сертификата

Диаграмма вариантов использования (use-case diagram) отражает функционал, доступный каждой группе пользователей и определяет назначение моделируемой информационной системы [3].

Диаграмма вариантов использования представлена на рисунке 2.3.

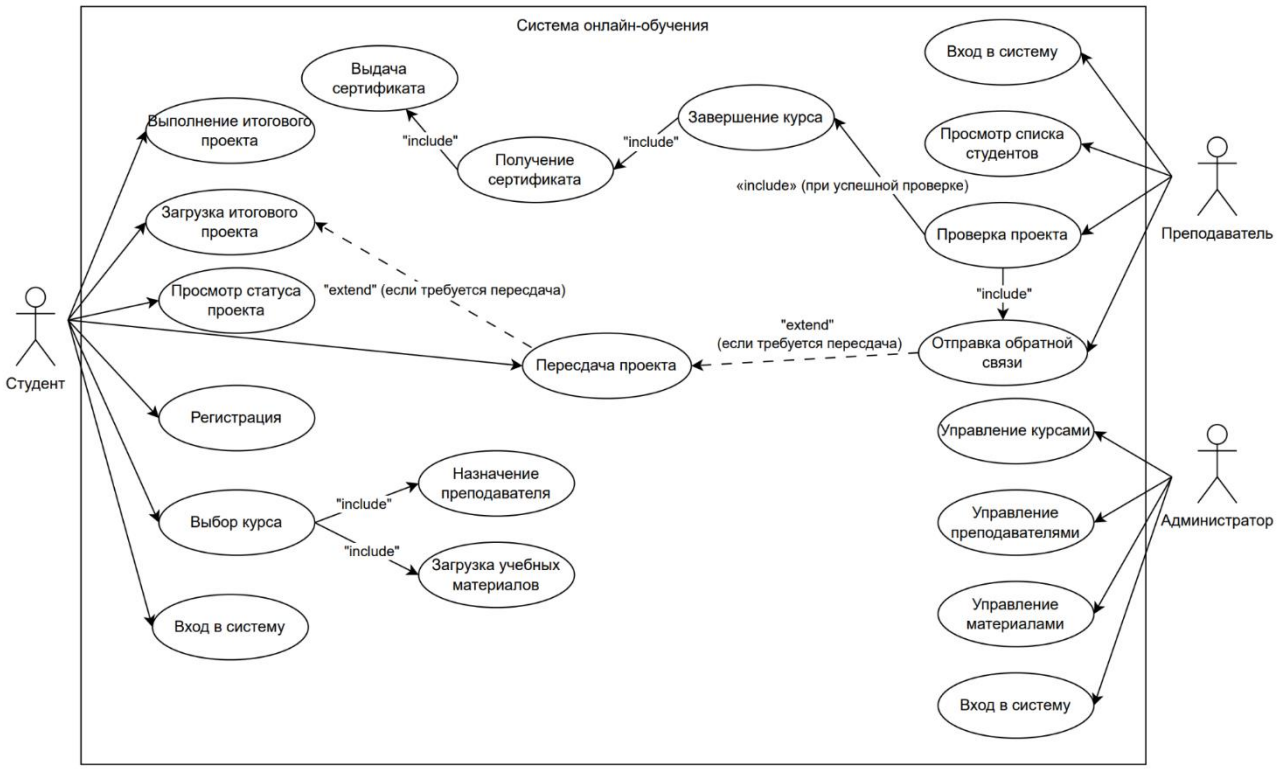

Рисунок 2.3 – Диаграмма вариантов использования

Источник: собственная разработка

Для примера детального описания варианта использования по стандарту шаблона RUP был выбран сценарий Business Use Case «Регистрация студента».

Краткое описание: данный Business Use Case позволяет пользователю создать учётную запись в системе онлайн-обучения для дальнейшего доступа к образовательным материалам и функционалу платформы.

Роли: Студент.

Стартовые условия**:** пользователь находится на странице регистрации.

Основной сценарий:

1. Пользователь открывает форму регистрации.

2. Пользователь вводит личные данные: ФИО, адрес электронной почты, пароль.

3. Пользователь подтверждает согласие с политикой обработки персональных данных.

4. Пользователь нажимает кнопку «Создать аккаунт».

5. Система проверяет корректность введённых данных.

6. Система создаёт учётную запись и отправляет письмо для подтверждения email.

7. Пользователь подтверждает регистрацию по ссылке в письме.

8. Система активирует аккаунт.

9. Регистрация завершена.

Альтернативный сценарий:

5. Система выявляет ошибки в заполненных данных.

- Система уведомляет пользователя о некорректности введённых данных (некорректный email, слабый пароль и т.д.).
- Пользователь исправляет данные и повторно отправляет форму регистрации.
- Возврат к шагу 5 основного сценария.

После определения всех требований к системе, формализации вариантов использования и построения диаграммы Use Case можно переходить к разработке архитектуры информационной системы. В следующих разделах будут представлены диаграммы, описывающие структуру системы и взаимодействие её компонентов.

ГЛАВА 3
СТРУКТУРА СИСТЕМЫ

Структура системы представляет собой совокупность взаимосвязанных элементов, которые обеспечивают целостность системы и позволяют ей функционировать как единое целое. Важно отметить, что правильное проектирование структуры системы является основой для достижения поставленных целей и обеспечения стабильности работы системы. В данном контексте под структурой системы понимаются как логические связи между модулями, так и механизмы взаимодействия компонентов, что, в свою очередь, способствует достижению высоких показателей качества работы системы.

Целью формирования структуры данной информационной системы является создание и описание комплекса элементов, которые обеспечат стабильную работу автоматизированной платформы для онлайн-курсов, а также её интеграцию с внешними и внутренними процессами. Структура системы будет определять организацию всех процессов, от регистрации студентов до выдачи сертификатов, обеспечивая взаимодействие всех участников (студентов, преподавателей, администраторов) через единую платформу.

Для более детального понимания работы системы, была разработана диаграмма действий, отображающая динамику потоков управления, которые могут быть связаны с несколькими вариантами использования системы. Диаграмма действий представляет собой поведенческую диаграмму, иллюстрирующую последовательность шагов и действий, которые выполняются в рамках конкретного бизнес-процесса. Использование диаграммы действий позволяет наглядно визуализировать рабочие процессы, а также анализировать последовательность выполнения операций в системе.

Для более точного моделирования процессов в языке UML применяются диаграммы активности (activity diagrams). Эти диаграммы служат своего рода блок-схемами, которые описывают последовательность выполнения операций во времени и отображают динамические аспекты поведения системы. Каждый элемент на диаграмме активности соответствует выполнению отдельной операции или действия, и переход в следующее состояние происходит только по завершению операции, выполненной на предыдущем шаге. Диаграммы активности в UML могут использоваться для моделирования бизнес-процессов, изучения потоков событий и определения требований к функциональности системы.

В контексте языка UML деятельность (activity) представляет собой совокупность вычислений или операций, выполняемых системой, где результат каждой операции или действия может быть изменением состояния системы или

возвращением конкретного значения. Для системы, диаграмма активности описывает последовательность действий, таких как регистрация студента, выбор курса, загрузка материалов, выполнение итогового проекта и получение сертификата. Каждый из этих шагов вносит свой вклад в общую работу системы и взаимодействие её компонентов.

Разработанная диаграмма активности, которая описывает ключевые бизнес-процессы системы, представлена на рисунке 3.1.

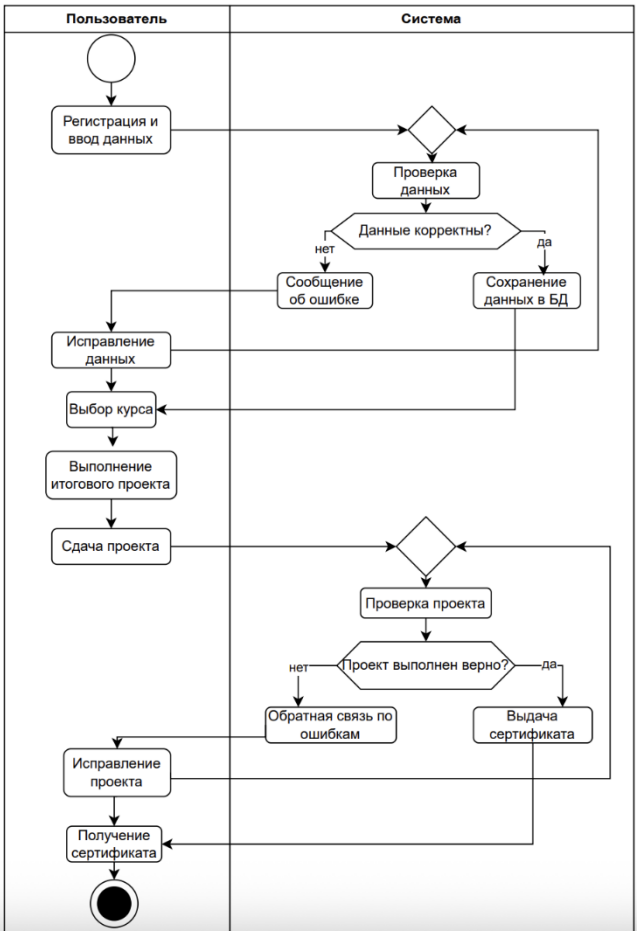

Рисунок 3.1 – Диаграмма активности системы онлайн-курсов

Источник: собственная разработка

В данной главе была рассмотрена структура информационной системы и подробно представлена диаграмма активности, которая иллюстрирует последовательность взаимодействий студентов, преподавателей и системы. На основе этих данных будет продолжено проектирование модельных и проектных классов автоматизированной информационной системы, что позволит перейти к следующему этапу разработки — созданию архитектуры системы.

ГЛАВА 4
ПРОЕКТИРОВАНИЕ КЛАССОВ

Данная глава посвящена выделению объектов проектируемой автоматизированной информационной системы онлайн-курсов, определению концептуальных классов, установлению связей между ними, уточнению их атрибутов и дальнейшему формированию диаграмм классов. Результатом является создание структурной модели предметной области, отражающей статические элементы системы и их взаимосвязи.

Диаграмма концептуальных классов (class diagram) служит для представления статической структуры системы в терминах объектно-ориентированного подхода. Она демонстрирует основные сущности предметной области, связи между ними и их ключевые характеристики. На данном этапе диаграмма не отражает временные аспекты функционирования, а представляет основу для дальнейшего проектирования логики и поведения системы.

Каждый класс на диаграмме отображает концепт, обладающий состоянием (атрибутами) и поведением (методами). Имя класса указывается в верхней части блока, за ним следуют атрибуты и операции. В ходе анализа требований и вариантов использования были выделены сущности, которые образуют концептуальную модель системы онлайн-курсов.

На основе анализа предметной области сформирован перечень концептуальных классов, представленный в таблице 4.1.

Таблица 4.1 — Выделенные концептуальные классы

Класс	**Основание для выделения**
Student	Пользователь, проходящий обучение
Teacher	Сущность, выполняющая проверку проектов
Course	Образовательная программа
Material	Учебные материалы курса
Project	Итоговая работа студента
Certificate	Документ о завершении курса

Окончание таблицы 4.1

StudentDB	Хранилище данных о студентах
CourseDB	Хранилище данных о курсах и материалах
TeacherDB	Хранилище преподавателей
ProjectDB	Хранилище итоговых работ
CertificateDB	Хранилище сведений о сертификатах
SystemController	Система, осуществляющая автоматизацию процессов

Источник: собственная разработка

На основе анализа вариантов использования были определены связи-ассоциации между сущностями. Они отражают взаимодействие и обмен информацией в процессе работы системы.

Ассоциация (association) представляет собой структурную связь между объектами классов. Она показывает, каким образом сущности предметной области взаимодействуют между собой при выполнении бизнес-процессов.

Связи, выявленные при моделировании, приведены в таблице 4.2.

Таблица 4.2 — Связи-ассоциации концептуальных классов

Название связи	**Выход**	**Вход**	**Категория**	**Назначение**
Проходит курс	Student	Course	Communicate	Студент выбирает и обучается на курсе
Назначается	Teacher	Course	Communicate	Курс закреплён за преподавателем
Содержит материалы	Course	Material	Communicate	Курс содержит набор учебных материалов
Выполняет проект	Student	Project	Communicate	Студент создаёт итоговую работу

Окончание таблицы 4.2

Проверяет	Teacher	Project	Communicate	Преподаватель оценивает проект
Получает сертификат	Student	Certificate	Communicate	Студент получает сертификат после завершения
Хранит данные	StudentDB	Student	Communication	База данных содержит записи о студентах
Хранит курсы	CourseDB	Course	Communication	База данных содержит информацию о курсах
Хранит преподавателей	TeacherDB	Teacher	Communication	БД преподавателей
Хранит проекты	ProjectDB	Project	Communication	БД проектов студентов
Хранит сертификаты	CertificateDB	Certificate	Communication	БД сертификатов
Управляет данными	System Controller	Все БД	Control	Система загружает, обновляет и фиксирует данные

Источник: собственная разработка

Атрибуты классов определяются на основе информационных требований сценариев. Атрибут должен иметь простой тип (строка, число, дата). Если данные имеют сложную структуру, они выносятся в отдельный класс и связываются посредством ассоциации.

Ниже представлены основные атрибуты, которые включаются в концептуальные классы:

- **Student**: id, name, email, password, currentCourse
- **Teacher**: id, name, specialization

- **Course**: id, title, description, difficulty
- **Material**: id, type, filename, uploadDate
- **Project**: id, studentId, courseId, status, grade
- **Certificate**: id, studentId, courseId, issueDate

На основе выделенных сущностей, их свойств и взаимосвязей была построена диаграмма концептуальных классов, которая представлена на рисунке 4.1. Она отражает структуру системы онлайн-курсов и взаимодействие основных объектов.

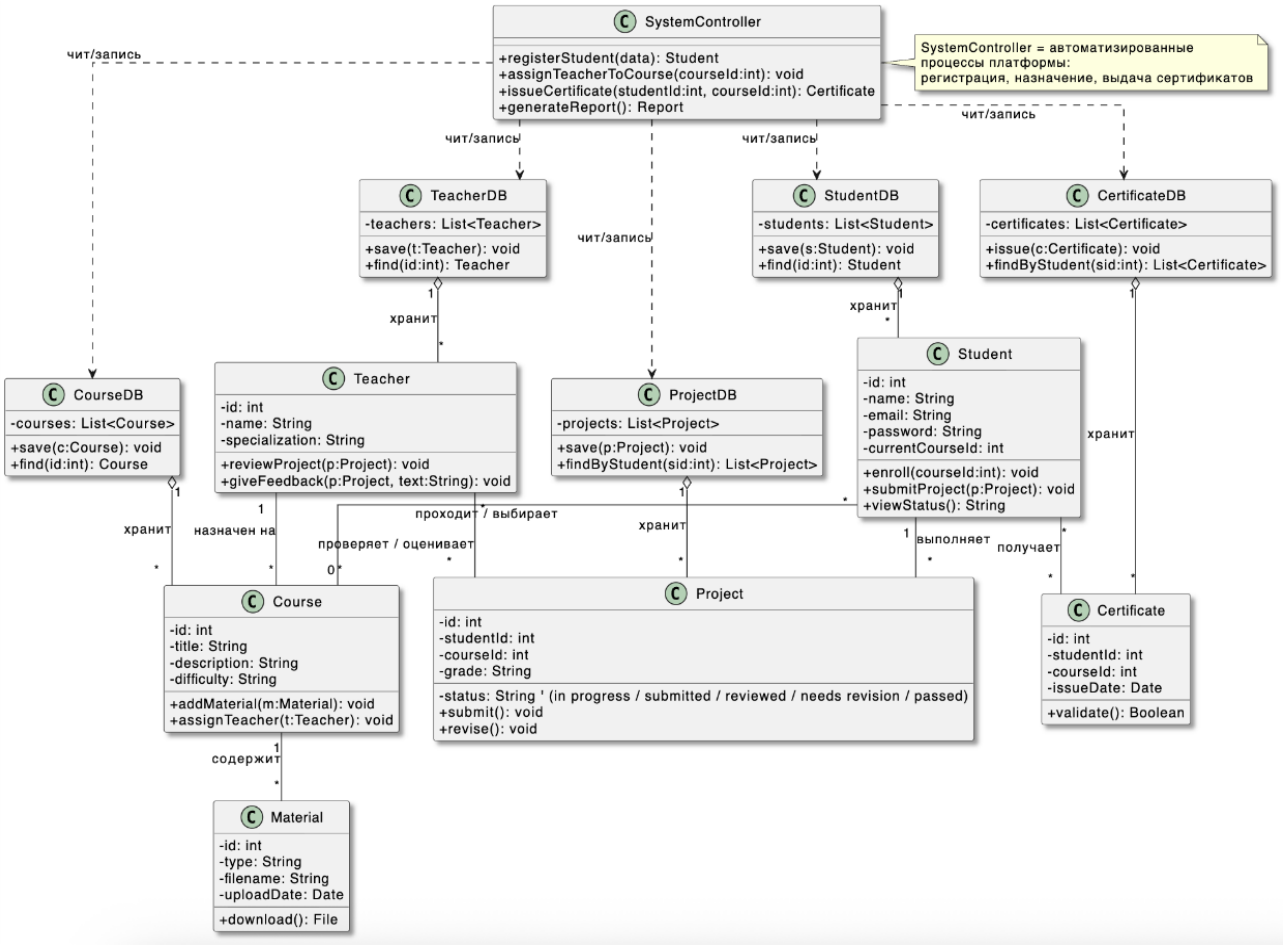

Рисунок 4.1 – Диаграмма концептуальных классов

Источник: собственная разработка

Диаграммы взаимодействия используются для моделирования поведения системы через обмен сообщениями между объектами, участвующими в выполнении определённого процесса. Такие диаграммы позволяют отобразить порядок вызовов, реакции системы и взаимодействие сущностей при реализации конкретного варианта использования.

Они применяются как для моделирования поведения отдельных операций, так и для описания целых вариантов использования. В качестве участников могут выступать объекты классов, подсистемы, компоненты, действующие лица и другие типы классификаторов.

Диаграммы взаимодействий имеют две основные формы:

1. **диаграммы последовательности**, отображающие процесс во временной оси;

2. **диаграммы коммуникации**, акцентирующие структуру связей между объектами.

В рамках данной работы рассмотрим построение диаграммы для варианта использования «Запись пользователя на онлайн-курс». Данный процесс является одним из ключевых для всей образовательной платформы, потому что именно он связывает интерес пользователя с механизмами внутренней обработки заявок, оплатой, формированием доступа и дальнейшим обучением. По сути, этот сценарий отражает путь студента от выбора курса до момента, когда система фиксирует его намерение учиться и создаёт формальную заявку в базе данных. Для корректного выполнения операции задействуются интерфейсные формы, модули валидации данных, компоненты работы с БД и сервисы уведомлений.

Наименование: запись на онлайн-курс в АИС.

Краткое описание: данный вариант использования позволяет пользователю выбрать курс, ознакомиться с информацией, заполнить необходимые данные и оформить запись на курс через автоматизированную информационную систему.

Стартовые условия: пользователь должен быть зарегистрирован или авторизован в системе.

Основной сценарий

1. Пользователь открывает страницу выбора курса «Course Catalog».

2. Система отображает список доступных курсов.

3. Пользователь выбирает конкретный курс.

4. Система открывает карточку курса «Course info».

5. Пользователь нажимает «Записаться».

6. Система открывает форму записи «Enrollment form».

7. Пользователь вводит необходимые данные (контакты, способ оплаты, дополнительные параметры).

8. Система загружает введённые сведения в соответствующую базу данных.

9. Пользователь подтверждает запись.

10. Система формирует заявку, сохраняет её в БД и отображает уведомление об успешной записи.

На рисунке 4.2 представлена диаграмма взаимодействий для варианта использования «Запись на онлайн-курс», отражающая обмен сообщениями между пользователем, интерфейсными формами и компонентами системы.

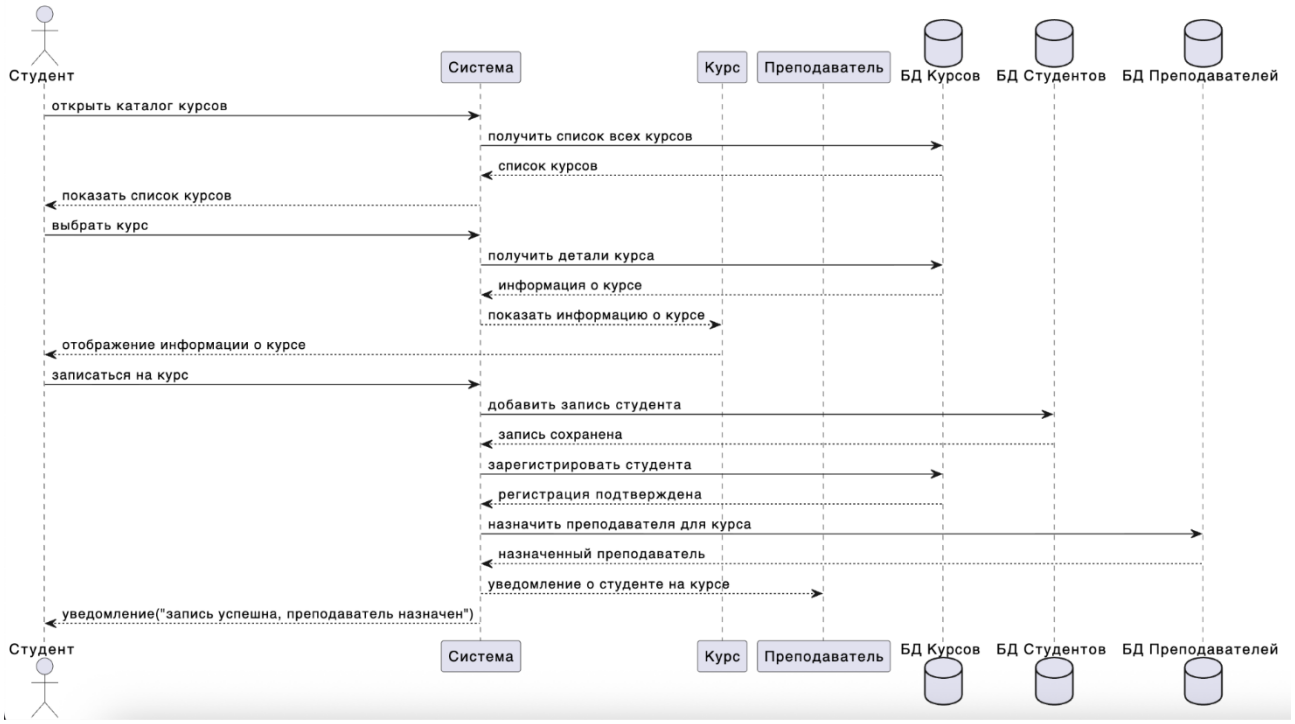

Рисунок 4.2 – Диаграмма взаимодействий

Источник: собственная разработка

Сценарии вариантов использования и диаграммы взаимодействия позволяют описать последовательность операций, которые выполняются системой и участниками процесса при обучении студента на онлайн-курсе. Однако для анализа логики работы отдельных ключевых объектов требуется зафиксировать их поведение вне контекста остальных элементов системы. Для этого используется диаграмма состояний, показывающая жизненный цикл объекта, возможные состояния и переходы между ними.

Диаграмма состояний представляет собой ориентированный граф, вершинами которого являются состояния объекта, а переходы (дуги) определяют условия, при которых происходит смена состояния. Подобное представление основано на формальной модели конечного автомата, в которой каждый объект может находиться только в одном состоянии в конкретный момент времени.

Состояние может содержать следующие элементы:

- **entry** — действие, выполняемое при входе в состояние;
- **exit** — действие при выходе;
- **do** — основная деятельность объекта внутри состояния.

В рамках проектируемой автоматизированной информационной системы была разработана диаграмма состояний ключевого объекта «Итоговый проект студента». Данная диаграмма описывает возможные изменения состояния проекта на этапах подготовки, проверки, доработки и финального утверждения преподавателем. Состояния итогового проекта представлены в таблице 4.3.

Таблица 4.3 — Состояния итогового проекта

Состояние	Характеристика состояния
Создан	entry/create — система инициирует объект проекта после выбора студентом курса
В работе	do/edit — студент выполняет проект, загружает материалы, редактирует данные
Отправлен на проверку	do/sendForReview — студент отправляет проект преподавателю
На проверке	do/check — преподаватель выполняет оценку проекта
Требует доработки	exit/requestChanges — преподаватель оставляет замечания и комментарии
Пересдан	do/resubmit — студент повторно загружает исправленный проект
Проверен и утверждён	do/approve — преподаватель подтверждает успешное выполнение проекта
Отклонён	exit/reject — преподаватель отклоняет проект при несоответствии требованиям
Завершён	финальное состояние — жизненный цикл проекта закрыт, данные зафиксированы в БД

Источник: собственная разработка

Таблица 4.4 формализует поведение объекта «Итоговый проект» и устраняет неоднозначности при реализации переходов между состояниями в системе.

Таблица 4.4 — Спецификация переходов между состояниями

Выход	Вход	Переход / Условие
Создан	В работе	Студент приступает к выполнению итогового задания

Окончание таблицы 4.4

В работе	Отправлен на проверку	Студент завершает работу и отправляет проект преподавателю
Отправлен на проверку	На проверке	Преподаватель открывает проект для оценки
На проверке	Требует доработки	Преподаватель оставляет комментарии и отмечает ошибки
Требует доработки	Пересдан	Студент вносит исправления и повторно загружает проект
Пересдан	На проверке	Преподаватель повторно принимает проект на проверку
На проверке	Проверен и утверждён	Преподаватель подтверждает корректность и качество выполненного задания
На проверке / Отправлен на проверку	Отклонён	Преподаватель отвергает проект при грубых нарушениях или несоответствии критериям
Проверен и утверждён	Завершён	Система фиксирует результаты, выдаёт студенту сертификат
Отклонён	Завершён	Жизненный цикл проекта закрывается без выдачи сертификата

Источник: собственная разработка

Спецификация переходов определяет строгую последовательность состояний, через которые проходит объект, и фиксирует правила перехода между ними. Такой подход позволяет формализовать бизнес-логику, устранить неоднозначности при внедрении функционала и обеспечить единое понимание процессов между разработчиками, аналитиками и тестировщиками. Благодаря этому диаграмма служит основой для проверки корректности поведения системы, упрощает разработку модулей и помогает выявлять потенциальные ошибки ещё на этапе проектирования.

На диаграмме состояний (рисунок 4.3) представлен полный жизненный цикл объекта «Итоговый проект», начиная с его создания системой и заканчивая финальным утверждением или отклонением.

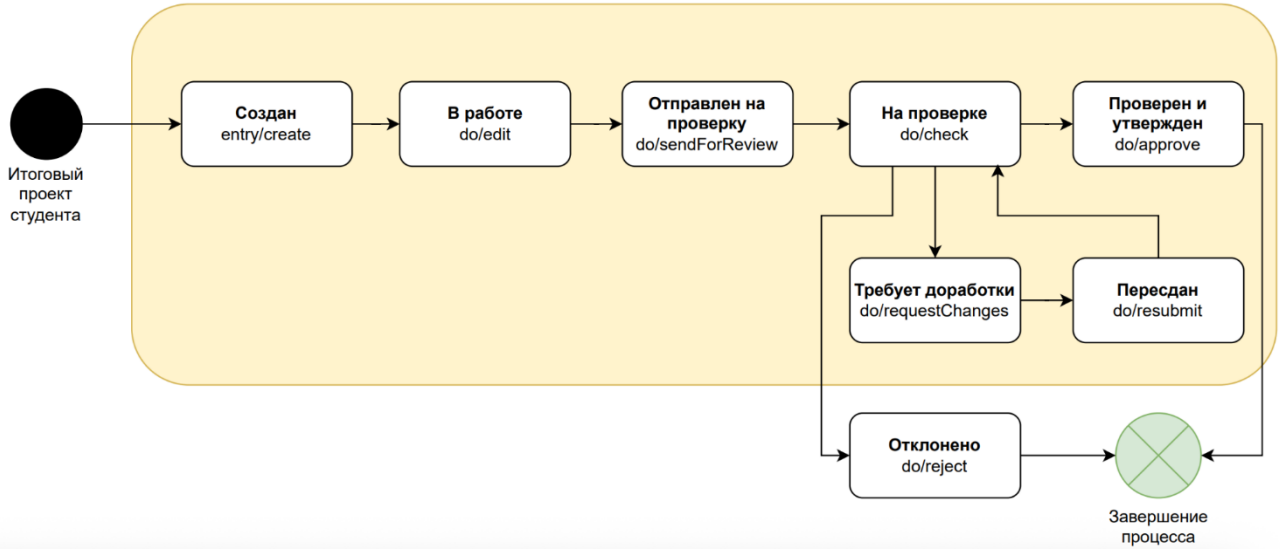

Рисунок 4.3 – Диаграмма состояние

Источник: собственная разработка

Процесс начинается со состояния «Создан», в котором система формирует карточку проекта сразу после выбора студентом курса. Далее студент переходит к выполнению задания, и проект находится в состоянии «В работе». После завершения работы пользователь отправляет проект преподавателю, что переводит объект в состояние «Отправлен на проверку».

Когда преподаватель открывает проект для оценки, он переходит в состояние «На проверке». Если выявлены недостатки, преподаватель запрашивает доработку и переводит проект в состояние «Требует доработки», после чего студент может загрузить исправленный вариант, в результате чего проект переходит в состояние «Пересдан».

После повторной проверки преподаватель может:

- утвердить проект, переводя его в состояние «Проверен и утверждён»;
- отклонить проект, после чего он получает статус «Отклонён».

Завершением жизненного цикла является переход в финальное состояние «Завершён», после чего состояние объекта не изменяется, а данные фиксируются в базе данных.

Диаграмма проектных классов предназначена для представления внутренней структуры разрабатываемой информационной системы онлайн-курсов в терминах программных классов и связей между ними.

Каждый класс отображается в виде прямоугольника, разделённого на три секции:

- имя класса,

● атрибуты, описывающие данные, хранящиеся в объекте,

● операции (методы), реализующие поведение объекта.

В отличие от диаграммы концептуальных классов, проектная диаграмма ориентирована на реализацию и отражает архитектурные решения системы. Она учитывает:

● наличие сервисных и контроллерных классов,

● структуру хранения данных,

● типизацию атрибутов,

● операции, выделенные на основе диаграмм взаимодействий и состояний,

● реальное распределение обязанностей между модулями системы.

На основании построенной ранее диаграммы взаимодействий (рис. 4.2) и диаграммы состояний итогового проекта (рис. 4.3) были уточнены методы, выполняемые каждым классом. На следующем этапе был определён набор атрибутов, необходимых для хранения учебных материалов, учётных данных пользователей, итоговых проектов и сертификатов. Итоговый перечень атрибутов проектных классов представлен в таблице 4.4.

Таблица 4.4 – Атрибуты проектных классов

Класс	**Атрибут**	**Назначение**
Student	studentID	Уникальный идентификатор студента
	name	Имя и фамилия студента
	email	Адрес электронной почты
	passwordHash	Хеш пароля
	currentCourseID	Идентификатор выбранного курса
Teacher	teacherID	Уникальный идентификатор преподавателя
	name	ФИО преподавателя
	specialization	Направление преподавания
Course	courseID	Уникальный идентификатор курса

Продолжение таблицы 4.4

	title	Название курса
	description	Описание курса
	difficulty	Уровень сложности
Material	materialID	Уникальный идентификатор учебного материала
	type	Тип (видео, PDF, презентация)
	filename	Имя файла
	uploadDate	Дата загрузки
Project	projectID	Идентификатор итогового проекта
	studentID	Студент, выполняющий проект
	courseID	Курс, к которому относится проект
	status	Текущее состояние проекта
	grade	Итоговая оценка преподавателя
Certificate	certificateID	Идентификатор сертификата
	studentID	Студент-владелец сертификата
	courseID	Курс, завершённый студентом
	issueDate	Дата выдачи сертификата
StudentDB	connection	Параметры подключения к базе студентов
	records	Записи о студентах

Окончание таблицы 4.4

TeacherDB	connection	Параметры подключения к базе преподавателей
	records	Записи о преподавателях
CourseDB	connection	Подключение к базе курсов
	records	Информация о курсах и материалах
ProjectDB	connection	Подключение к базе проектов
	records	Хранение проектов и комментариев
CertificateDB	connection	Подключение к базе сертификатов
	records	Записи о выданных сертификатах
SystemController	controllerID	Идентификатор сервиса
	logLevel	Режим логирования
	mode	Режим работы системы

Источник: собственная разработка

На основе выделенных атрибутов и ранее определённых операций была построена диаграмма проектных классов, демонстрирующая структуру разрабатываемой системы. Она включает:

1. Классы предметной области

- Student
- Teacher
- Course
- Material
- Project
- Certificate

2. Сервисные классы

- SystemController — отвечает за автоматизацию действий: назначение преподавателя, выдачу сертификата, загрузку материалов.

● Базы данных: StudentDB, TeacherDB, CourseDB, ProjectDB, CertificateDB

3. Типы связей

● **Ассоциации** между Student–Course, Student–Project, Teacher–Project

● **Агрегация** Course → Material

● **Зависимости** контроллера от всех баз данных

● **Односторонние ссылки** Student → Certificate

Комбинация этих элементов формирует архитектурный каркас системы, обеспечивая поддержку всех бизнес-процессов: регистрация студента, назначение курса, загрузка материалов, выполнение проекта, проверка преподавателем и выдача сертификата.

Диаграмма проектных классов на рисунке 4.4 демонстрирует, как бизнес-объекты и сервисные компоненты взаимодействуют между собой при выполнении учебного процесса и формируют основу для дальнейшей реализации в виде классов выбранного языка программирования.

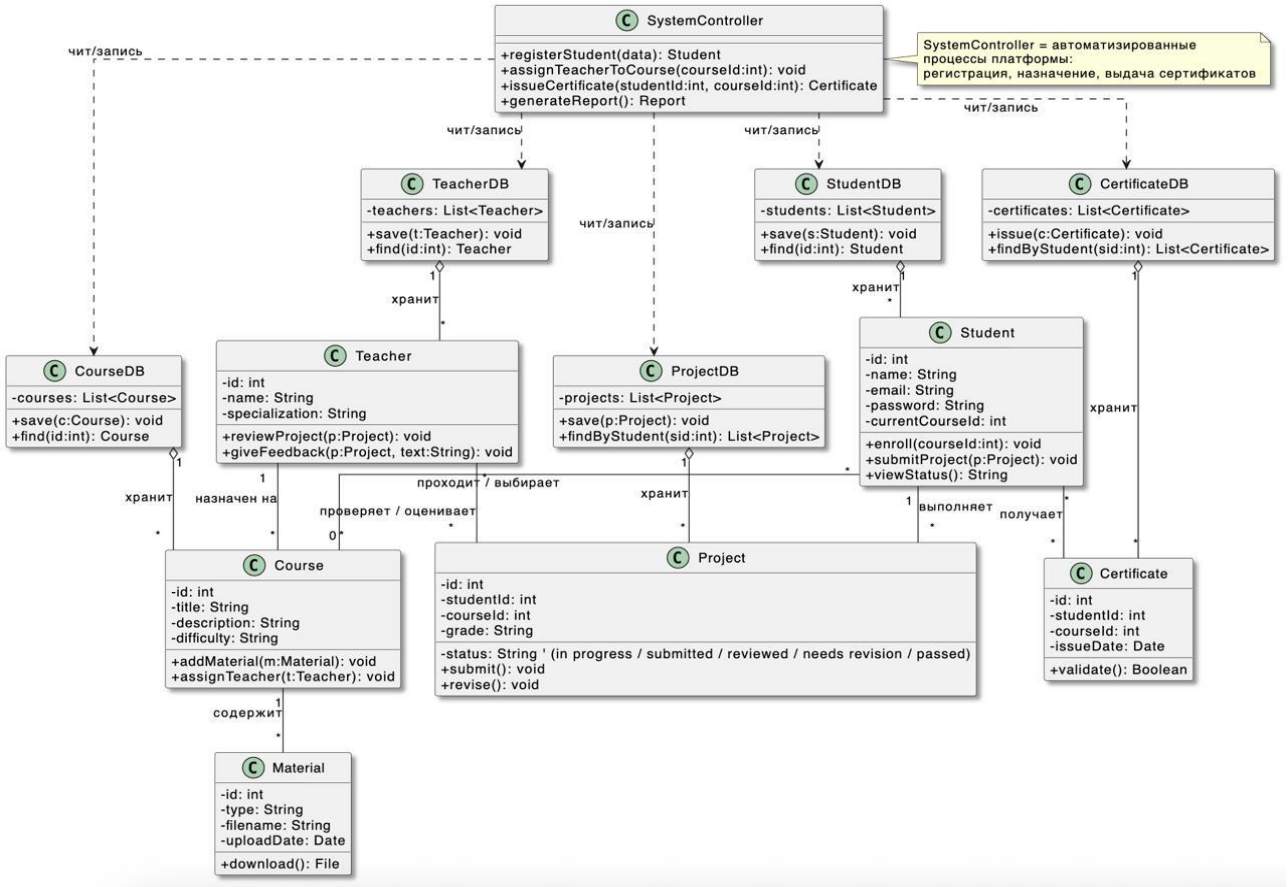

Рисунок 4.4 – Диаграмма проектных классов

Источник: собственная разработка

ГЛАВА 5
ПРОЕКТИРОВАНИЕ СХЕМЫ ДАННЫХ

На предыдущих этапах были определены концептуальные и проектные классы системы, что позволяет перейти к формированию объектной модели данных и схемы хранения информации.

Объектная модель данных представляет собой графическое описание структуры базы данных, отражающее основные сущности, их атрибуты и связи между ними. Эта модель демонстрирует, какие данные используются в системе онлайн-курсов, каким образом они взаимосвязаны и какие операции над ними выполняются. Объектная модель позволяет понять внутреннюю организацию хранилища и служит основой для построения физической схемы данных.

В состав объектной модели базы данных системы онлайн-курсов входят следующие сущности:

- **Student** – данные о студентах (регистрация, авторизация, участие в курсах).
- **Teacher** – информация о преподавателях и их специализациях.
- **Course** – данные о курсах и их характеристиках.
- **Material** – учебные материалы, прикреплённые к курсам.
- **Project** – итоговые работы студентов, их статус и результаты проверки.
- **Certificate** – данные о выданных сертификатах после завершения курса.
- **соответствующие базы данных (таблицы):** StudentDB, CourseDB, TeacherDB, ProjectDB, CertificateDB.

Объектная модель базы данных представлена на рисунке 5.1.

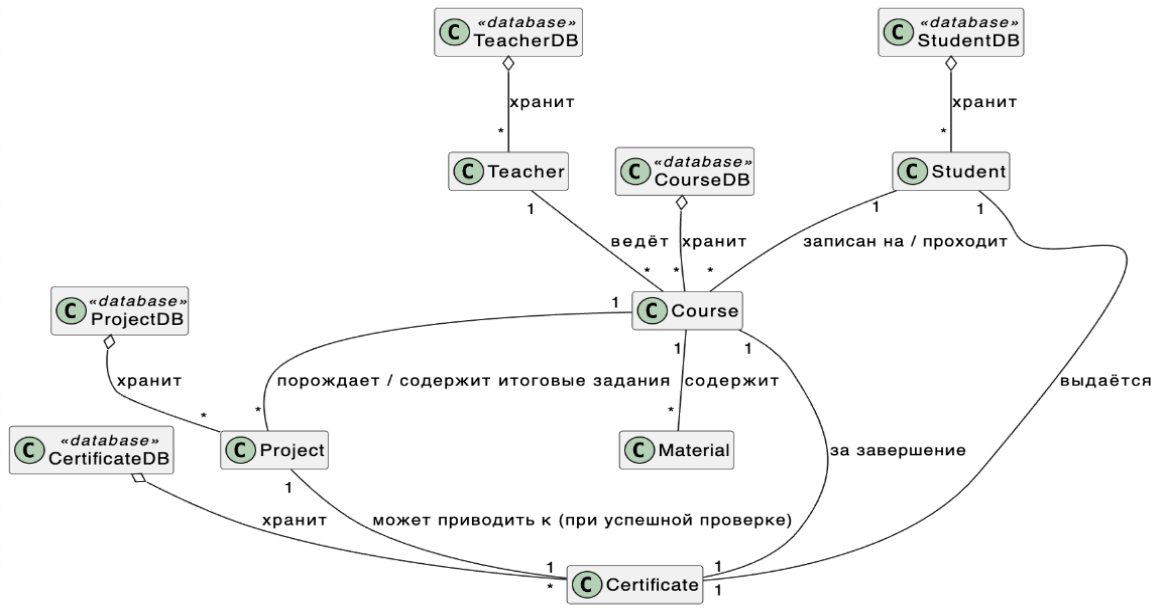

Рисунок 5.1 – Объектная модель базы данных системы онлайн-курсов

Источник: собственная разработка

На основе объектной модели создаётся схема данных, которая определяет структуру таблиц, типы полей, первичные и внешние ключи, связи «один-ко-многим» и «многие-ко-многим». Схема данных служит основой для физической реализации базы данных в выбранной СУБД.

Схема данных учитывает:

- необходимость хранения регистрационных данных пользователя;
- хранение курсов и связанных с ними материалов;
- назначение преподавателей;
- сохранение итоговых проектов и истории их проверки;
- формирование сертификатов после завершения курса.

Схема данных представлена на рисунке 5.2.

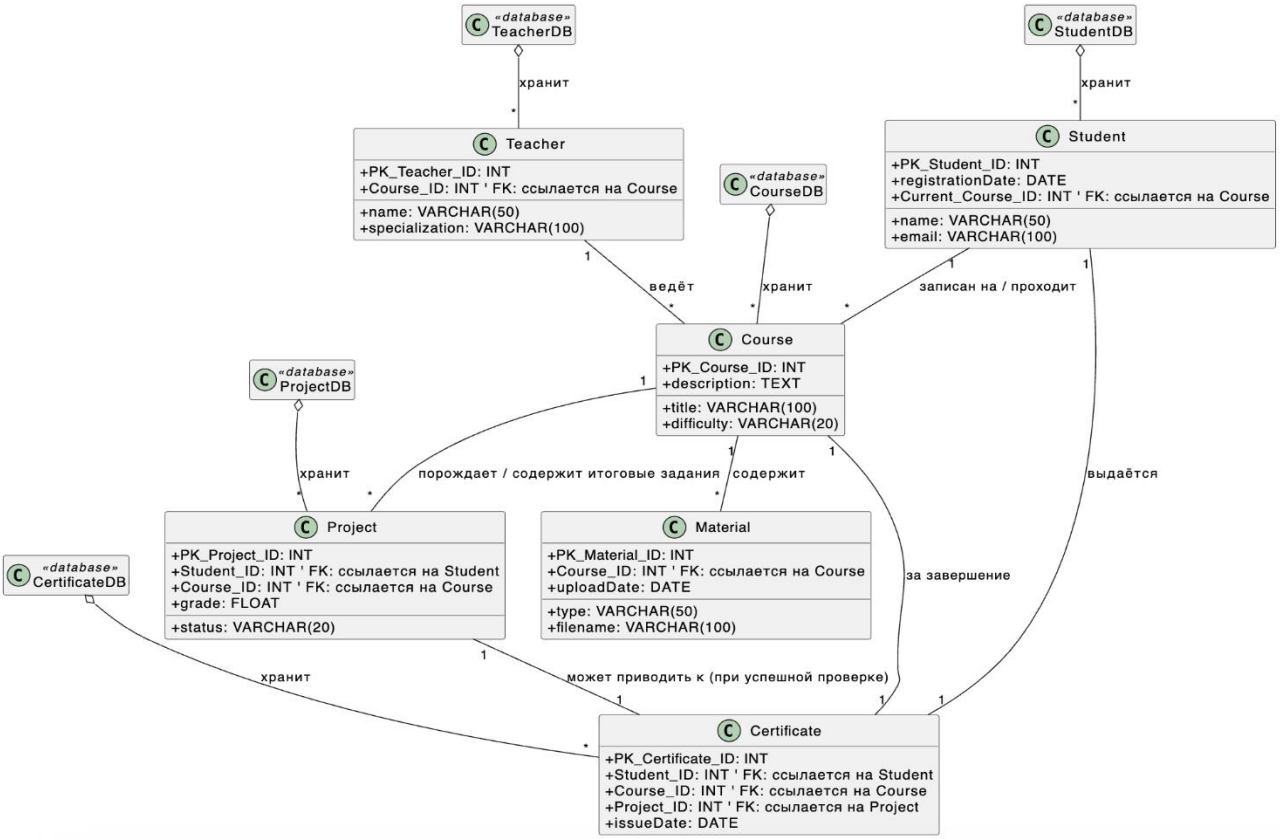

Рисунок 5.2 – Схема данных автоматизированной информационной системы онлайн-курсов

Источник: собственная разработка

В процессе проектирования схемы данных были разработаны элементы серверной логики, обеспечивающие контроль целостности и автоматизацию работы с базой данных. К ним относятся триггеры и хранимые процедуры.

Поскольку итоговый проект является ключевым элементом образовательного процесса, важно предотвратить некорректное изменение идентификаторов проектов. Для защиты от случайного удаления или модификации идентификатора была создана серверная логика в виде триггера.

Триггер запрещает изменение значения **projectID** в таблице *Project*, чтобы исключить риск потери данных и нарушения связей с другими таблицами.

```
CREATE TRIGGER trg_protectProjectID
ON Project
FOR UPDATE AS
IF UPDATE(projectID)
BEGIN
   PRINT 'Запрещено изменять идентификатор projectID'
   ROLLBACK TRAN
END
GO
```

Для удобства преподавателей и администраторов была разработана хранимая процедура, которая по номеру итогового проекта выводит сведения о студенте и статусе выполнения работы.

```
CREATE PROC prProjectInfo
   @ProjectID int
AS
SELECT
   Student.name AS StudentName,
   Course.title AS CourseTitle,
   Project.status AS ProjectStatus
FROM Project
JOIN Student ON Project.studentID = Student.studentID
JOIN Course ON Project.courseID = Course.courseID
WHERE Project.projectID = @ProjectID
GO
```

Процедура позволяет быстро получить структурированную информацию о ходе выполнения проекта и облегчает работу преподавателей при проверке итоговых заданий.

Файл, содержащий SQL-код создания всех таблиц, первичных и внешних ключей, триггеров и хранимых процедур, формируется автоматически при помощи системы проектирования (например, Enterprise Architect). Этот файл позволяет воссоздать всю структуру базы данных, описанную в схеме данных, и служит основой для последующей реализации АИС онлайн-обучения.

Листинг SQL-кода, сгенерированного по итогам проектирования схемы данных, приведён в Приложении Б.

Разработанная схема данных полностью отражает функциональные требования информационной системы онлайн-курсов и обеспечивает хранение всей необходимой информации для автоматизации процесса обучения. В

дальнейшем она будет использоваться при реализации серверной части приложения, создании API и интеграции системы с пользовательским интерфейсом.

ГЛАВА 6
РАЗРАБОТКА ШАБЛОНА КОДА

Одним из свойств CASE-среды Enterprise Architect является возможность генерации программного кода, реализующего разработанную модель системы. В данной главе рассматривается процесс формирования диаграммы компонентов автоматизированной информационной системы онлайн-курсов и разработка шаблонов программного кода классов на языке C++ на основе этой диаграммы.

Сформированная ранее диаграмма проектных классов (глава 4) была использована в качестве основы для построения диаграммы компонентов. На диаграмме компонентов отражаются основные структурные элементы системы (модули, формы, контроллеры, классы предметной области), а также связи между ними.

Для упрощения архитектуры и повышения наглядности диаграмма компонентов была разбита на три логических пакета в соответствии с шаблоном архитектуры **boundary–control–entity**:

● **entity** – классы предметной области, представляющие данные (Student, Teacher, Course, Project, Certificate и др.);

● **boundary** – классы интерфейса взаимодействия с пользователем (веб-страницы, формы регистрации, формы выбора курса, формы загрузки проектов и просмотра сертификатов);

● **control** – классы, реализующие бизнес-логику и управление процессами (контроллеры записи на курс, проверки проектов, выдачи сертификатов и т.п.).

Разбиение диаграммы на пакеты представлено на рисунке 6.1.

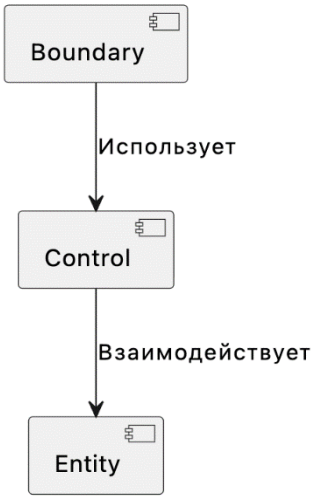

Рисунок 6.1 – Разбиение компонентов АИС онлайн-курсов на пакеты boundary, control и entity

Источник: собственная разработка

Пакет boundary содержит компоненты, обеспечивающие взаимодействие пользователя (студента, преподавателя, администратора) с системой. Каждый компонент представляет собой форму или веб-страницу, через которую выполняется отдельный вариант использования.

К основным компонентам пакета относятся:

- **RegistrationForm** – форма регистрации студента;
- **LoginForm** – форма авторизации;
- **CourseCatalogPage** – страница выбора курса;
- **CourseDetailsPage** – страница с информацией о курсе;
- **ProjectUploadPage** – форма загрузки итогового проекта;
- **CertificateViewPage** – страница просмотра сертификата.

Структура пакета boundary представлена на рисунке 6.2.

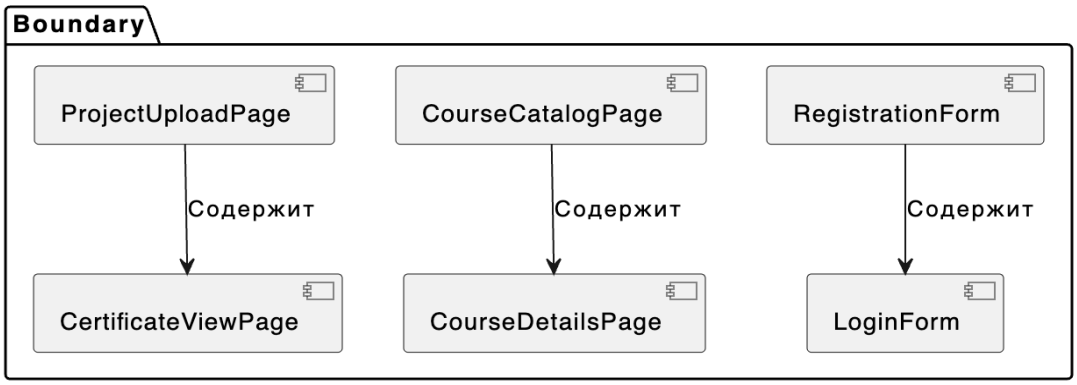

Рисунок 6.2 – Пакет boundary информационной системы онлайн-курсов

Источник: собственная разработка

Пакет entity включает классы предметной области, которые непосредственно отражают данные, хранящиеся в базе (см. главы 4 и 5). Эти классы используются как модельные сущности при работе контроллеров и интерфейсных форм.

К основным классам пакета относятся:

- **Student** – данные о студенте (ФИО, email, учетные данные, текущий курс);
- **Teacher** – данные о преподавателе и его специализации;
- **Course** – информация о курсе, описании и сложности;
- **Material** – учебные материалы курса;
- **Project** – итоговый проект студента, его статус, оценка;
- **Certificate** – сведения о выданных сертификатах.

Пакет entity также может включать классы-обёртки для работы с базами данных (например, StudentRepository, CourseRepository и др.), если это предусмотрено выбранной архитектурой.

Структура пакета entity представлена на рисунке 6.3.

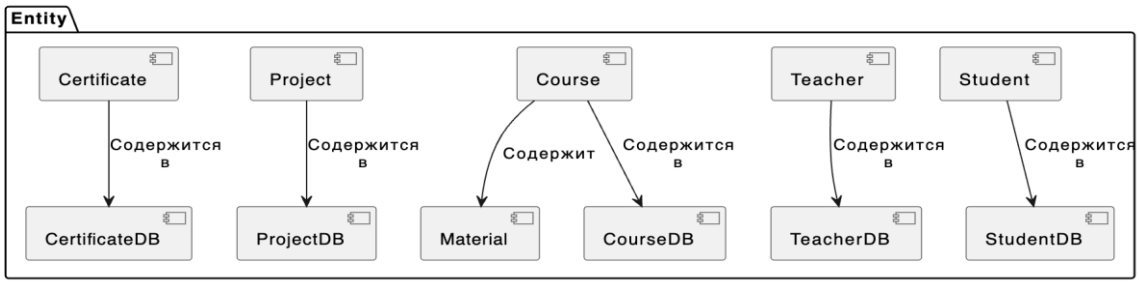

Рисунок 6.3 – Пакет entity информационной системы онлайн-курсов

Источник: собственная разработка

Пакет control содержит классы, реализующие бизнес-логику системы и координирующие взаимодействие между интерфейсными формами (boundary) и сущностями (entity). В рамках автоматизированной системы онлайн-курсов данный пакет обеспечивает реализацию ключевых бизнес-процессов: регистрация студента, запись на курс, выполнение и проверка итогового проекта, выдача сертификата.

К основным классам пакета control относятся:

- **AuthController** – управление регистрацией и авторизацией студентов и преподавателей;
- **EnrollmentController** – логика выбора курса и записи студента на обучение;
- **ProjectController** – управление жизненным циклом итогового проекта (создание, отправка на проверку, пересдача, утверждение);
- **CertificateController** – формирование и фиксация сертификатов при завершении курса;
- **NotificationService** – отправка уведомлений студентам и преподавателям (о проверке проекта, назначении преподавателя, выдаче сертификата).

Структура пакета control представлена на рисунке 6.4.

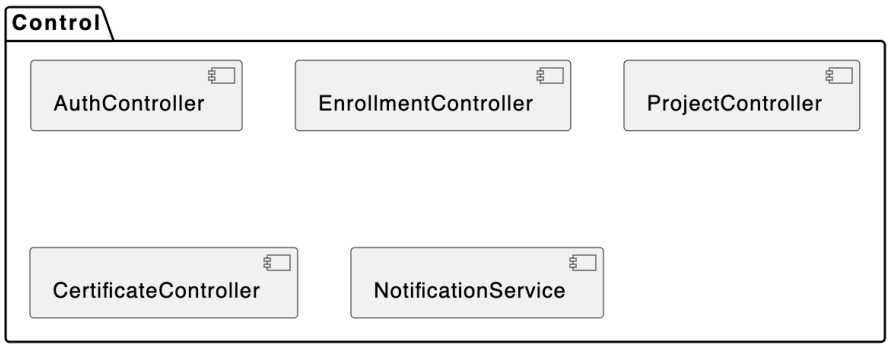

Рисунок 6.4 – Пакет control информационной системы онлайн-курсов

Источник: собственная разработка

На основе рассмотренных пакетов была сформирована итоговая диаграмма компонентов, которая отражает взаимосвязи между boundary-, control- и entity-компонентами. На диаграмме показано, какие интерфейсные формы используют определённые контроллеры, а контроллеры, в свою очередь, обращаются к сущностям предметной области и репозиториям данных.

Итоговая диаграмма компонентов информационной системы онлайн-курсов представлена на рисунке 6.5.

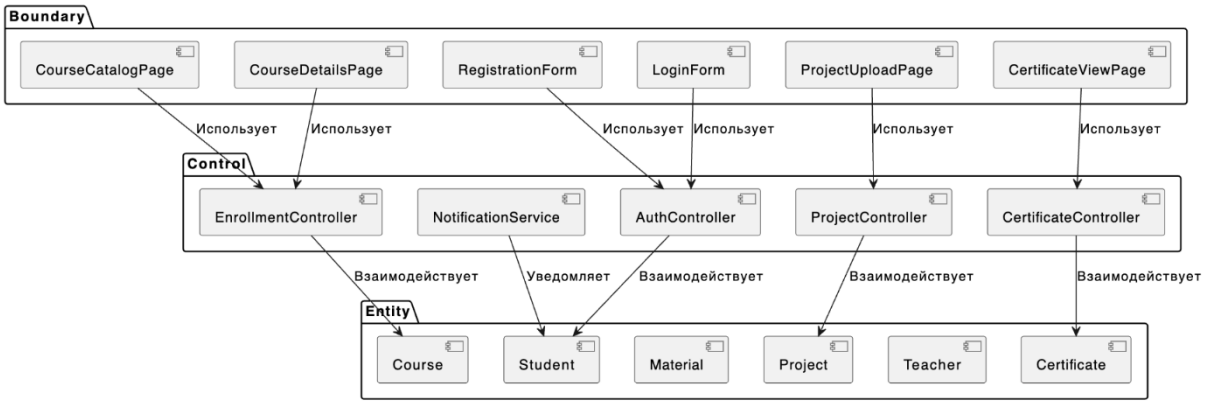

Рисунок 6.5 – Диаграмма компонентов АИС онлайн-курсов

Источник: собственная разработка

После формирования диаграмм компонентов в среде Enterprise Architect был выполнен этап генерации программного кода. Для каждого класса, входящего в пакеты boundary, control и entity, автоматически создаются файлы заголовков и реализации (header и source файлы) на языке C++. Эти файлы представляют собой шаблоны, содержащие объявления классов, их атрибутов и методов, соответствующих разработанной UML-модели.

В качестве примера приведён фрагмент сгенерированного кода для класса интерфейсного уровня **EnrollmentForm**, обеспечивающего запись студента на онлайн-курс. Данный класс относится к пакету boundary и взаимодействует с контроллером EnrollmentController и сущностями Student и Course.

```
#include "Student.h"
#include "Course.h"
#include "EnrollmentController.h"

class EnrollmentForm
{
public:
   EnrollmentForm();
   virtual ~EnrollmentForm();
```

```
    // Открытие формы выбора курса
    void open();

    // Установка активного студента
    void setStudent(Student* student);

    // Установка выбранного курса
    void setCourse(Course* course);

    // Подтверждение записи на курс
    void submit();

private:
    Student* m_student;
    Course*  m_course;
    EnrollmentController* m_controller;

    // Вспомогательные данные интерфейса
    int selectedCourseID;
};
```

Аналогичным образом генерируются шаблоны кода для других классов, например:

- **Student**, **Course**, **Project**, **Certificate** (пакет entity);
- **AuthController**, **ProjectController**, **CertificateController** (пакет control);
- **RegistrationForm**, **LoginForm**, **ProjectUploadPage**, **CertificateViewPage** (пакет boundary).

Сгенерированный код содержит базовые структуры классов, соответствующие атрибуты и объявления методов. На следующем этапе разработки программист может дополнить эти шаблоны реализацией бизнес-логики, обработкой ошибок, взаимодействием с СУБД и пользовательским интерфейсом.

Разработка и генерация шаблонов кода на основе диаграммы компонентов позволяет:

- обеспечить соответствие программной реализации разработанной архитектурной модели;
- сократить трудозатраты на создание базовых классов и их структур;
- снизить риск несоответствия между документацией и фактической реализацией;

● ускорить процесс разработки серверной и клиентской логики системы.

Сгенерированные файлы могут быть использованы в дальнейшем как основа для расширения функциональности автоматизированной информационной системы онлайн-курсов, интеграции с базой данных, веб-интерфейсом и внешними сервисами.

ГЛАВА 7
ОРГАНИЗАЦИЯ ТЕСТИРОВАНИЯ И ПРИЁМКИ ИНФОРМАЦИОННОЙ СИСТЕМЫ

В данной главе рассматривается организация тестирования автоматизированной информационной системы онлайн-курсов. Тестирование является важнейшим этапом жизненного цикла разработки программного обеспечения и используется для проверки корректности работы системы, её соответствия функциональным требованиям и устойчивости к ошибкам.

Согласно определению, тестирование — это процесс сопоставления фактического поведения программного продукта с ожидаемым, основанным на заранее подготовленных тестовых сценариях и критериях качества [14]. Цель тестирования — убедиться, что система способна обеспечивать выполнение всех необходимых функций, предусмотренных техническим заданием.

Для оценки качества работы информационной системы обычно применяются различные виды тестирования. В контексте автоматизированной системы онлайн-курсов ключевое значение имеют:

- функциональное тестирование;
- тестирование пользовательских сценариев;
- тестирование интерфейсов;
- smoke test (приемочное тестирование);
- critical path test (проверка критического пути).

В данной главе особое внимание уделено двум базовым видам проверки: smoke-тестированию и тестированию критического пути, которые определяют готовность системы к эксплуатации.

Smoke-тестирование проводится для быстрой проверки основных модулей и функций системы. Его задача — убедиться, что ключевые компоненты корректно взаимодействуют между собой, отображаются основные страницы интерфейса и доступны основные операции пользователя [16, 18].

Для системы онлайн-курсов к перечню проверяемых функций относятся:

- доступность стартовой страницы платформы;
- работа форм регистрации и авторизации;
- загрузка каталога курсов;
- открытие карточки курса;
- доступ к личному кабинету студента;
- отображение учебных материалов.

Smoke-тестирование позволяет убедиться, что система в целом функционирует корректно и готова к более детальной проверке.

Тестирование критического пути направлено на проверку ключевых сценариев, которые отражают основной процесс взаимодействия пользователя с системой [17, 18]. Для образовательных платформ это особенно важно, так как цикл прохождения курса должен выполняться безошибочно.

Для автоматизированной системы онлайн-курсов критический путь включает следующие последовательные операции:

1. регистрация нового студента;
2. вход в личный кабинет;
3. выбор курса в каталоге;
4. запись на обучение;
5. получение учебных материалов;
6. выполнение и загрузка итогового проекта;
7. обработка проекта преподавателем;
8. получение результата проверки;
9. формирование сертификата после успешного завершения курса.

Проверка критического пути позволяет убедиться, что система поддерживает последовательность основных действий без логических ошибок и некорректных переходов.

В рамках проверки системы выделяются ключевые элементы, требующие обязательной оценки качества:

- валидация данных при регистрации и авторизации;
- корректность отображения курсов и информации о них;
- создание, сохранение и изменение статусов итогового проекта;
- обработка ошибок при загрузке файлов;
- назначение преподавателя и отображение данных в его интерфейсе;
- корректное формирование сертификата и его отображение в личном кабинете;
- взаимодействие с базами данных, включая StudentDB, CourseDB, TeacherDB, ProjectDB, CertificateDB.

Проверка данных объектов позволяет оценить соответствие системы требованиям, определённым в техническом задании.

На основании анализа требований, схемы данных, диаграмм вариантов использования и проектных решений формируются критерии, позволяющие оценить готовность системы к внедрению. К ним относятся:

- наличие всех ключевых функций, описанных в техническом задании;
- корректность прохождения основных пользовательских сценариев;
- соответствие структуры данных разработанной модели;
- устойчивость к ошибкам при выполнении базовых операций;
- удобство и логичность интерфейса.

Структура проведённой проверки подтверждает готовность системы к эксплуатации при условии реализации программных модулей в соответствии с моделью.

В главе была рассмотрена организация тестирования автоматизированной информационной системы онлайн-курсов. Определены основные виды проверки, объекты тестирования и ключевые пользовательские сценарии. На основе результатов анализа можно сформировать план дальнейшей приёмки системы и подготовки её к внедрению.

ЗАКЛЮЧЕНИЕ

В ходе разработки автоматизированной информационной системы для организации онлайн-курсов была проведена всесторонняя работа по проектированию и моделированию всех ключевых компонентов системы. Процесс проектирования включал в себя тщательное определение функциональных и нефункциональных требований, создание моделей данных и бизнес-процессов, а также проектирование архитектуры системы.

Основное внимание было уделено автоматизации ключевых процессов, таких как регистрация студентов, выбор курсов, назначение преподавателей, выполнение итоговых проектов, их оценка и выдача сертификатов. Эти процессы были тщательно описаны и представлены в виде диаграмм бизнес-процессов и диаграмм вариантов использования, что позволило четко определить взаимодействие пользователей и системы.

Для хранения и обработки данных была спроектирована структура базы данных, включающая несколько взаимосвязанных таблиц, таких как базы данных студентов, преподавателей, курсов, проектов и сертификатов. Все данные обрабатываются и сохраняются в соответствии с проектной схемой, что обеспечивает эффективность и надежность системы.

Проектирование системы компонентов позволило создать модульную архитектуру, разделённую на пакеты boundary, control и entity, что способствует улучшению управления и развитию системы в будущем. В частности, компоненты взаимодействуют через интерфейсы и контроллеры, обеспечивая гибкость и масштабируемость системы.

Тестирование показало, что система функционирует корректно в рамках базовых сценариев использования, что подтверждает её готовность к эксплуатации. Тестирование критического пути обеспечило уверенность в том, что система будет корректно работать в реальных условиях эксплуатации и справляться с основными процессами образовательной платформы.

Таким образом, разработанная информационная система онлайн-курсов соответствует всем поставленным целям и требованиям, обеспечивая надежное и эффективное управление обучением на платформе. Внедрение данной системы значительно повысит качество образовательного процесса, улучшит взаимодействие студентов и преподавателей, а также ускорит процессы записи, оценки и выдачи сертификатов. Продолжение работы над проектом может включать расширение функционала, а также улучшение интерфейса и повышения уровня безопасности данных пользователей, что позволит системе оставаться актуальной и востребованной в условиях постоянного развития технологий и требований пользователей.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кальницкий, В. В. «Основы проектирования информационных систем» / В. В. Кальницкий. — М.: КНОРУС, 2019. — 304 с.

2. Гальперин, С. А. «Проектирование программных систем» / С. А. Гальперин. — СПб.: БХВ-Петербург, 2020. — 380 с.

3. Ткачев, А. В. «Реализация UML в системах проектирования» / А. В. Ткачев. — М.: ДМК Пресс, 2017. — 220 с.

4. Гостев, В. А. «Автоматизация бизнес-процессов» / В. А. Гостев. — М.: Инфра-М, 2016. — 416 с.

5. Воронцов, И. В. «Моделирование и анализ информационных систем» / И. В. Воронцов. — М.: КНОРУС, 2018. — 312 с.

6. Шевченко, О. М. «Основы разработки и эксплуатации информационных систем» / О. М. Шевченко. — М.: Высшая школа, 2020. — 272 с.

7. Ларман, К. «UML: Объектно-ориентированное моделирование» / К. Ларман. — 3-е изд. — М.: Вильямс, 2019. — 688 с.

8. Пономарев, В. С. «Основы проектирования систем управления» / В. С. Пономарев. — СПб.: Питер, 2019. — 512 с.

9. Александров, Д. М. «Разработка и проектирование баз данных» / Д. М. Александров. — М.: Диалектика, 2017. — 384 с.

10. Рябов, Д. С. «Практическое руководство по тестированию программного обеспечения» / Д. С. Рябов. — М.: Издательский дом "Питер", 2019. — 424 с.

11. Фролова, Н. В. «Управление проектами в ИТ-сфере» / Н. В. Фролова. — СПб.: Питер, 2020. — 280 с.

12. Вейд, Р. «Программирование с использованием UML» / Р. Вейд. — М.: БХВ-Петербург, 2017. — 312 с.

13. Клименко, В. К. «Основы программирования для информационных систем» / В. К. Клименко. — М.: Логос, 2018. — 256 с.

14. Диаграммы взаимодействия [Электронный ресурс] – 2022. – Режим доступа: https://studfile.net/preview/4084877/page:16/ – Дата доступа: 29.11.2025.

15. Дергачев, И. О. «Технологии и методы разработки информационных систем» / И. О. Дергачев. — М.: КНОРУС, 2020. — 288 с.

16. Евтушенко, В. П. «Методы проектирования программных систем» / В. П. Евтушенко. — СПб.: БХВ-Петербург, 2021. — 364 с.

ПРИЛОЖЕНИЕ А

Техническое задание «ПРОЕКТИРОВАНИЕ АВТОМАТИЗИРОВАННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ОРГАНИЗАЦИИ ОБУЧЕНИЯ НА ОНЛАЙН-КУРСАХ»

наименование организации - разработчика ТЗ на АС

УТВЕРЖДАЮ

Руководитель (должность, наименование предприятия - заказчика АС)

Личная подпись Расшифровка подписи

Печать

Дата

УТВЕРЖДАЮ

Руководитель (должность, наименование предприятия - разработчик” ИС)

Личная подпись Расшифровка подписи

Печать

Дата

Автоматизированная информационная система организации обучения на онлайн-курсах

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

На 8 листах

Действует с 10.09.2025

1. Общие сведения

1.1. Наименование системы

1.1.1. Полное наименование системы: Автоматизированная информационная система организации обучения на онлайн-курсах.

1.1.2. Краткое наименование системы: АИС «Онлайн-обучение».

1.2. Основания для проведения работ

Работа выполняется в рамках курсового проекта и основывается на учебном задании, согласованном с условным Заказчиком — образовательной компанией ООО «EduTech Solutions».

1.3. Наименование организаций – Заказчика и Разработчика

1.3.1. Заказчик: ООО «EduTech Solutions». Адрес: г. Минск, ул. Богдановича, 120. Телефон: +375 (29) 555-44-33.

1.3.2. Разработчик: Родионова Алеся Олеговна. Адрес: г. Минск, ул. Платонова, 18. Телефон: +375 (25) 777-21-08.

1.4. Плановые сроки начала и окончания работы

Начало работ: 15.09.2025. Окончание работ: 20.12.2025. Сроки могут корректироваться по согласованию с Заказчиком или в соответствии с учебным планом.

1.5. Источники и порядок финансирования

Финансирование проекта осуществляется за счёт средств Заказчика — ООО «EduTech Solutions». Порядок финансирования определяется внутренними договорённостями.

1.6. Порядок оформления и предъявления заказчику результатов работ

Разработчик поэтапно предоставляет результаты создания АИС согласно учебному плану. По завершении этапов формируются отчетные документы: описание требований, диаграммы и проектные решения. Испытания выполняются в соответствии с Программой и методикой испытаний. По окончании работ Исполнитель передает Заказчику комплект проектной документации на разработанную систему.

2. Назначение и цели создания системы

2.1. Назначение системы

Автоматизированная информационная система предназначена для организации и сопровождения процесса онлайн-обучения, обеспечивая взаимодействие студентов, преподавателей и автоматических сервисов платформы. В рамках АИС автоматизируются следующие процессы:

— Регистрация и авторизация пользователей;

— Управление курсами и учебными материалами;

— Назначение преподавателей и взаимодействие со студентами;

— Выполнение и загрузка итоговых проектов;

— Проверка работ, выставление оценок и предоставление обратной связи;

— Формирование сертификатов о завершении обучения;

— Ведение баз данных студентов, курсов, проектов и результатов обучения;

— Генерация уведомлений и отчетов о ходе обучения.

2.2. Цели создания ИС

Информационная система создается с целью:

• автоматизации ключевых процессов онлайн-обучения;

• повышения удобства взаимодействия между студентами и преподавателями;

• обеспечения оперативного доступа к учебным материалам;

• оптимизации процесса проверки итоговых работ;

• повышения прозрачности и точности учета результатов обучения;

• ускорения формирования и выдачи электронных сертификатов.

3. Характеристика объектов автоматизации

Таблица А.1 – Характеристика объектов автоматизации

Структурное подразделение	**Наименование процесса**	**Возможность автоматизации**	**Решение об автоматизации в процессе**
Отдел образовательных технологий	Регистрация студентов	Возможна	Будет автоматизировано
Отдел учебных материалов	Загрузка материалов курса	Возможна	Будет автоматизировано
Отдел преподавателей	Назначение преподавателей	Возможна	Будет автоматизировано
Отдел тестирования	Проверка проектов студентов	Возможна	Будет автоматизировано

Источник: собственная разработка

4. Требования к системе

4.1. Требования к системе в целом

4.1.1. Требования к структуре и функционированию

Система должна быть реализована по архитектуре «клиент–сервер», центральная база данных хранит информацию о пользователях, курсах, материалах и итоговых проектах, работа системы обеспечивается круглосуточно с краткими регламентными перерывами.

4.1.2. Требования к численности и квалификации персонала

Для эксплуатации АИС требуются: руководитель подразделения – 1 чел., администратор контента – 2 чел., администратор БД – 2 чел., администратор отчетности – 1 чел. Администратор БД должен владеть СУБД, резервным копированием и оптимизацией; администратор отчетности — методами аналитики, SQL и инструментами визуализации; руководитель — понимать работу обучающих систем и современные технологические решения; администратор контента — уметь работать с мультимедиа, процессами загрузки материалов и обеспечивать их безопасность. Режимы работы персонала определяются регламентами заказчика.

4.1.3. Показатели назначения

Данные должны храниться не менее 5 лет, одновременно системой могут пользоваться около 300 студентов и до 20 преподавателей. Время отклика интерфейса — до 5 секунд, формирование аналитических отчетов — до 2 минут. Приспособляемость обеспечивается администрированием, возможностью обновления модулей, изменением процедур доступа и параметров отображения данных. Система должна сохранять работоспособность при перебоях электроснабжения до 10 минут, временной потере связи до 10 минут с последующей синхронизацией, выходе из строя сервера БД с уведомлением администраторов, а обновление ПО должно выполняться с минимальными перерывами в доступе.

4.1.4. Требования к надежности

Надежность обеспечивается применением аппаратных и программных средств, регулярным администрированием и соблюдением правил эксплуатации. Возможные аварийные ситуации: сбои питания серверов и рабочих станций, неисправности сети, ошибки ПО и непредвиденные сбои серверных компонентов. Проверка надежности проводится на этапах проектирования, тестирования и эксплуатации.

4.1.5. Требования к эргономике и эстетике

Интерфейс должен быть типизированным, русскоязычным, единообразным по структуре и оформлению; для частых действий предусмотрены «горячие» клавиши; при ошибках отображаются понятные сообщения с рекомендациями. Навигационные элементы, графические значки и текстовые обозначения унифицированы.

4.1.6. Требования к эксплуатации и обслуживанию

Обслуживание оборудования и ПО выполняется согласно документации производителей.

4.1.7. Требования к защите информации

Система защищается программно-техническими и организационными средствами, защита действует на всех этапах обработки данных. Права доступа разрабатываются по принципу «запрещено всё, что не разрешено». Антивирус установлен на всех рабочих местах.

4.1.8. Требования по сохранности информации при авариях

Обязательно резервное копирование данных, выход из строя дисков не должен приводить к потере работоспособности или утрате информации.

4.1.9. Требования к защите от внешних воздействий

Система должна стабильно функционировать в пределах температурных и влажностных режимов, указанных производителями оборудования, и при допустимых вибрациях.

4.1.10. Требования по стандартизации

Разработка должна использовать методологии IDEF0, DFD, ER-моделирование в инструментах BPWin и ERWin.

4.2. Требования к функциям системы

Система должна обеспечивать сбор, обработку, хранение и предоставление данных о пользователях, курсах, учебных материалах, итоговых проектах и результатах обучения. Пользователям предоставляются актуальные данные: студентам — материалы и задания, преподавателям — работы студентов и функции проверки, администрации — аналитика и отчеты. Функции работают постоянно, с возможностью оперативного изменения регламентов обработки данных.

4.3. Требования к видам обеспечения

4.3.1. Математическое обеспечение — не требуется.

4.3.2. Информационное обеспечение. Данные должны быть точными, актуальными и непротиворечивыми. Поддерживаемые форматы входных данных: CSV, JSON, XML. Используются справочники пользователей, курсов, преподавателей и типов материалов. Они поддерживают добавление, редактирование, удаление и просмотр, а также механизмы целостности и интеграции с внешними системами.

4.3.3. Лингвистическое обеспечение. Используются языки C#, SQL и формат XML, интерфейс — русскоязычный.

4.3.4. Программное обеспечение: ОС Windows 10 или выше, архитектура x86/x64, СУБД Microsoft SQL Server 2008 или выше.

4.3.5. Техническое обеспечение: выделенные серверы заказчика с минимальными характеристиками CPU 4×2 ГГц, RAM 4 ГБ, HDD 500 ГБ, сеть

100 Мбит/с; рабочие станции операторов — CPU 1.8 ГГц, RAM 1 ГБ, HDD 250 ГБ, сеть 100 Мбит/с.

4.3.6. Метрологическое обеспечение — не требуется.

4.3.7. Организационное обеспечение должно обеспечивать выполнение сотрудниками всех автоматизированных и вспомогательных функций.

4.3.8. Методическое обеспечение — не требуется.

4.3.9. Патентная чистота: используемые технические и программные средства должны соответствовать лицензионным требованиям и обеспечивать патентную чистоту.

5. Состав и содержание работ по созданию системы

Работы по созданию АИС онлайн-обучения выполняются в три этапа:

1. Проектирование. Разработка эскизного проекта системы, создание технического проекта с определением структуры баз данных, интерфейсов пользователей и функциональных модулей (продолжительность — 1 месяц).
2. Разработка рабочей документации и программного обеспечения. Реализация модулей регистрации пользователей, управления курсами, загрузки учебных материалов, проверки итоговых проектов, формирования отчетов и выдачи сертификатов (продолжительность — 1 месяц).
3. Ввод системы в эксплуатацию. Настройка серверного оборудования и базы данных, тестирование работы системы, обучение пользователей и администраторов, сопровождение первых запусков платформы (продолжительность — 1 месяц).

6. Порядок контроля и приемки системы

6.1. Виды и объем испытаний системы

Система подвергается испытаниям следующих видов: предварительные испытания, опытная эксплуатация, приемочные испытания. Состав, объем и методы предварительных испытаний определяются документом «Программа и методика испытаний», разрабатываемым на стадии «Разработка рабочей документации». Состав, объем и методы опытной эксплуатации определяются документом «Программа опытной эксплуатации», разрабатываемым на стадии «Ввод системы в эксплуатацию». Состав, объем и методы приемочных испытаний определяются документом «Программа и методика испытаний», разрабатываемым на стадии «Ввод системы в эксплуатацию» с учетом результатов предварительных испытаний и опытной эксплуатации.

6.2. Требования к приемке работ по стадиям

Для приемки выполненных работ Исполнитель издает приказ о создании комиссии, в состав которой входит программа и методика испытаний АИС онлайн-обучения, обеспечивающая проверку функциональности, надежности,

безопасности и соответствия системы требованиям, установленным в техническом задании.

7. Требования к документированию

Перечень основных мероприятий:

- приведение поступающей в систему информации о пользователях, курсах, материалах и итоговых проектах к виду, пригодному для обработки системой;
- внесение необходимых изменений в функциональные модули платформы;
- создание условий функционирования АИС онлайн-обучения, обеспечивающих соответствие требованиям технического задания;
- создание необходимых подразделений и служб для сопровождения системы;
- определение сроков и порядка комплектования штата и обучения персонала.

Для обеспечения условий функционирования системы, соответствующих требованиям ТЗ, и возможности её эффективного использования, в организации Заказчика должен быть проведен комплекс мероприятий до начала этапа «Разработка рабочей документации и адаптация программ». В рамках этих мероприятий должны быть выполнены следующие работы:

- закупка и установка серверного и пользовательского оборудования;
- подготовка оборудования для размещения компонентов системы в соответствии с требованиями технического задания, настройка сетевой инфраструктуры и баз данных для корректной работы всех модулей платформы.

8. Источники разработки

На каждом этапе разработки разрабатываются и согласовываются Разработчиком, Родионовой Алесей Олеговной, и Заказчиком документы, соответствующие требованиям ГОСТ 34.602-2020. Все результаты работ по созданию системы и её модулей, по сборке и наладке отдельных подсистем и программных компонентов должны подробно документироваться. Документы, составленные на иностранных языках, должны иметь приложение с переводом на русский язык.

9. Источники разработки

Настоящее Техническое задание разработано на основе следующих документов и информационных материалов:

● Договор от 01.10.2025 между ООО «EduTech» и Родионовой Алесей Олеговной;

● ГОСТ 34.602-2020 — «Техническое задание на создание автоматизированной системы».

Настоящее ТЗ может дополняться и изменяться в процессе разработки и приемочных испытаний в установленном порядке по взаимному соглашению Заказчика и Разработчика

```
/* ---------------------------------------------------- */
/* Generated for Online Course System */
/* ---------------------------------------------------- */
/* Drop Foreign Key Constraints */
IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Student_Course]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Student] DROP CONSTRAINT [FK_Student_Course]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Project_Student]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Project] DROP CONSTRAINT [FK_Project_Student]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Project_Course]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Project] DROP CONSTRAINT [FK_Project_Course]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Certificate_Student]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Certificate] DROP CONSTRAINT [FK_Certificate_Student]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Certificate_Course]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Certificate] DROP CONSTRAINT [FK_Certificate_Course]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id =
object_id(N'[FK_Certificate_Project]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)
   ALTER TABLE [Certificate] DROP CONSTRAINT [FK_Certificate_Project]
GO

/* Drop Tables */
IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Student]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
   DROP TABLE [Student]
GO
```

```
IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Teacher]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [Teacher]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Course]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [Course]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Material]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [Material]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Project]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [Project]
GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object_id(N'[Certificate]') AND
OBJECTPROPERTY(id, N'IsUserTable') = 1)
    DROP TABLE [Certificate]
GO

/* Create Tables */
CREATE TABLE [Student] (
    [Student_ID] INT NOT NULL,
    [Name] VARCHAR(50) NULL,
    [Email] VARCHAR(100) NULL,
    [Current_Course_ID] INT NULL
)
GO

CREATE TABLE [Teacher] (
    [Teacher_ID] INT NOT NULL,
    [Name] VARCHAR(50) NULL,
    [Specialization] VARCHAR(50) NULL
)
```

```
GO

CREATE TABLE [Course] (
    [Course_ID] INT NOT NULL,
    [Title] VARCHAR(100) NULL,
    [Description] TEXT NULL,
    [Difficulty] VARCHAR(20) NULL
)
GO

CREATE TABLE [Material] (
    [Material_ID] INT NOT NULL,
    [Course_ID] INT NOT NULL,
    [Type] VARCHAR(50) NULL,
    [Filename] VARCHAR(100) NULL,
    [UploadDate] DATE NULL
)
GO

CREATE TABLE [Project] (
    [Project_ID] INT NOT NULL,
    [Student_ID] INT NOT NULL,
    [Course_ID] INT NOT NULL,
    [Status] VARCHAR(50) NULL,
    [Grade] FLOAT NULL
)
GO

CREATE TABLE [Certificate] (
    [Certificate_ID] INT NOT NULL,
    [Student_ID] INT NOT NULL,
    [Course_ID] INT NOT NULL,
    [Project_ID] INT NOT NULL,
    [IssueDate] DATE NULL
)
GO

/* Create Primary Keys */
ALTER TABLE [Student] ADD CONSTRAINT [PK_Student] PRIMARY KEY CLUSTERED
([Student_ID] ASC)
```

```
GO

ALTER TABLE [Teacher] ADD CONSTRAINT [PK_Teacher] PRIMARY KEY CLUSTERED
([Teacher_ID] ASC)
GO

ALTER TABLE [Course] ADD CONSTRAINT [PK_Course] PRIMARY KEY CLUSTERED
([Course_ID] ASC)
GO

ALTER TABLE [Material] ADD CONSTRAINT [PK_Material] PRIMARY KEY CLUSTERED
([Material_ID] ASC)
GO

ALTER TABLE [Project] ADD CONSTRAINT [PK_Project] PRIMARY KEY CLUSTERED
([Project_ID] ASC)
GO

ALTER TABLE [Certificate] ADD CONSTRAINT [PK_Certificate] PRIMARY KEY
CLUSTERED ([Certificate_ID] ASC)
GO

/* Create Foreign Key Constraints */
ALTER TABLE [Project] ADD CONSTRAINT [FK_Project_Student] FOREIGN KEY
([Student_ID]) REFERENCES [Student] ([Student_ID]) ON DELETE CASCADE ON UPDATE
CASCADE
GO

ALTER TABLE [Project] ADD CONSTRAINT [FK_Project_Course] FOREIGN KEY
([Course_ID]) REFERENCES [Course] ([Course_ID]) ON DELETE CASCADE ON UPDATE
CASCADE
GO

ALTER TABLE [Certificate] ADD CONSTRAINT [FK_Certificate_Student] FOREIGN KEY
([Student_ID]) REFERENCES [Student] ([Student_ID]) ON DELETE CASCADE ON UPDATE
CASCADE
GO
```

```
ALTER TABLE [Certificate] ADD CONSTRAINT [FK_Certificate_Course] FOREIGN KEY ([Course_ID]) REFERENCES [Course] ([Course_ID]) ON DELETE CASCADE ON UPDATE CASCADE
GO

ALTER TABLE [Certificate] ADD CONSTRAINT [FK_Certificate_Project] FOREIGN KEY ([Project_ID]) REFERENCES [Project] ([Project_ID]) ON DELETE CASCADE ON UPDATE CASCADE
GO

/* Create Stored Procedures */
CREATE PROCEDURE prStudentCourses
   @StudentID INT
AS
BEGIN
   SELECT Course.Title
   FROM Course
   JOIN Project ON Course.Course_ID = Project.Course_ID
   WHERE Project.Student_ID = @StudentID
END
GO

CREATE PROCEDURE prProjectStatus
   @ProjectID INT
AS
BEGIN
   SELECT Status
   FROM Project
   WHERE Project_ID = @ProjectID
END
GO
```